AD-A085 147 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/6 16/2

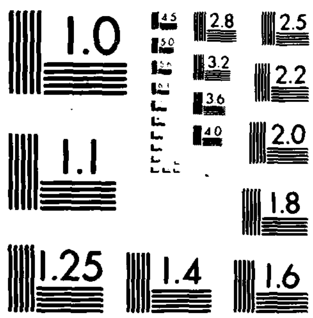
GUIDANCE AND CONTROL OF TACTICAL MISSILES.(U)

DEC 79 T A GROTE

UNCLASSIFIED NL

1 of 2

AD
A085147


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0

15

LEVEL II (2)

ADA085147

NAVAL POSTGRADUATE SCHOOL
Monterey, California

DTIC
ELECTE
JUN 5 1980
S A

THESIS

GUIDANCE AND CONTROL OF TACTICAL MISSILES

by

Thomas Alan Grote

December 1979

Thesis Advisor: D. J. Collins

Approved for public release; distribution unlimited

80 6 3 03

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO. AD-A085 147

3. RECIPIENT'S CATALOG NUMBER (9)

4. TITLE (and Subtitle) (6) Guidance and Control of Tactical Missiles.

5. TYPE OF REPORT & PERIOD COVERED Master's Thesis (December 1979)

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s) (10) Thomas Alan Grote

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93940

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Naval Postgraduate School
Monterey, California 93940

(11) 12. REPORT DATE December 1979

13. NUMBER OF PAGES 173

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)
Naval Postgraduate School
Monterey, California 93940
(12) 174

15. SECURITY CLASS. (of this report) Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Accession For
NTIS GRA&I
DDC TAB
Unannounced
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or special
A

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
MOD6DF - STM

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This thesis discusses the conversion of the MOD6DF computer program for use on the IBM-360 computer at the Naval Postgraduate School. The functioning program was modified to investigate the impact miss distance for the Supersonic Tactical Missile. When the initial y-displacement error exceeded 1800 feet, the missile did not acquire the target. All errors smaller than this resulted in miss distances within 0.5 feet of the target. The midcourse guidance reference altitude was changed to reflect a sea-skimming missile. This

251450

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
(Page 1) S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

simulation ran and impact was recorded. An attempt at adding random noise to the homing seeker was tried, but revealed that more information is required on this topic. The MOD6DF computer program was successfully converted and altered to run using the simplified ramjet model.

Approved for public release; distribution unlimited

Guidance and Control of Tactical Missiles

by

Thomas Alan Grote
Lieutenant, United States Navy
B.S.A.E., United States Naval Academy, 1974

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ENGINEERING SCIENCE

from

NAVAL POSTGRADUATE SCHOOL
December 1979

Author Thomas Alan Grote

Approved by: Daniel J Collins (Thesis Advisor)

Max F. Platzer
Chairman, Department of Aeronautical Engineering

William M. Tolles
Dean of Science and Engineering

ABSTRACT

This thesis discusses the conversion of the MOD6DF computer program for use on the IBM-360 computer at the Naval Postgraduate School. The functioning program was modified to investigate the impact miss distance for the Supersonic Tactical Missile. When the initial y-displacement error exceeded 1800 feet, the missile did not acquire the target. All errors smaller than this resulted in miss distances within 0.5 feet of the target. The midcourse guidance reference altitude was changed to reflect a sea-skimming missile. This simulation ran and impact was recorded. An attempt at adding random noise to the homing seeker was tried, but revealed that more information is required on this topic. The MOD6DF computer program was successfully converted and altered to run using the simplified ramjet model.

TABLE OF CONTENTS

LIST OF TABLES

<u>LIST OF FIGURES</u>

TABLE OF SYMBOLS AND ABBREVIATIONS

The following is a list of the abbreviations and some of the more common fortran symbols used in the MOD6DF computer program. Each symbol is defined in two ways. The primary source of identification is by its COMMON(3415) location and the second is by its fortran symbol. The fortran symbol is not always a good identifier since it will change from subroutine to subroutine (i.e. W and WE both are used for missile weight).

A. ABBREVIATIONS

Symbol	Definition
TP	tangent plane axes
BA	body axes
SA	stability axes

B. FORTRAN SYMBOLS

Symbol	Definition
TXBA(073)	missile thrust in the x-direction in BA (lb)
TYBA(074)	missile thrust in the y-direction in BA (lb)
TZBA(075)	missile thrust in the z-direction in BA (lb)
W(086)	missile weight (lb)
S(110)	missile reference area (ft^2)
CA(111)	drag coefficient
CY(113)	side-force coefficient
CZ(115)	normal force coeffieient
CBAR(116)	mean aerodynamic chord (ft)
CMQ(118)	damping in pitch coeffieient
CNR(119)	damping in yaw coefficient

Symbol	Definition
CLP(120)	damping in roll coefficient
CM(121)	pitching moment coefficient
CN(122)	yawing moment coefficient
CL(123)	rolling moment coefficient
CGI(136)	center of gravity (ft)
A(201)	moment-of-inertia about missile roll axis (x-body axis) (slug-ft^2)
B(202)	moment-of-inertia about missile pitch axis (y-body axis) (slug-ft^2)
CC(203)	moment-of-inertia about missile yaw axis (z-body axis) (slug-ft^2)
TSA(208)	angle between SA and BA (rad)
P(212)	missile angular velocity about x-BA (rad/s)
Q(216)	missile angular velocity about y-BA (rad/s)
R(220)	missile angular velocity about z-BA (rad/s)
AG(282)	unit conversion lb-slug
VXTP(286)	missile velocity in x-TP (ft/s)
XTP(290)	missile displacement in x-TP (ft)
VYTP(294)	missile velocity in y-TP (ft/s)
YTP(298)	missile displacement in y-TP (ft)
VZTP(302)	missile velocity in z-TP (ft/s)
ZTP(306)	missile displacement in z-TP (ft)
GZRO(404)	constant set to zero for flat earth gravitational field and set to one for a spherical gravitational field
ER(405)	angular velocity of earth (rad/s)
ALPO(406)	angle between north and x-TP (rad)

Symbol	Definition
OLAMO(407)	latitude origin of tangent plane
HO(414)	distance tangent plane is from earth (ft)
GO(415)	gravitational acceleration (ft/s^2)
HREF(501)	reference altitude for midcourse guidance (ft)
RE(503)	earth's radius (ft)
H(507)	altitude normal to earth (ft)
AMACH(520)	missile Mach number
THET(521)	missile pitch angle, TP (rad)
PSI(522)	missile yaw angle, TP (rad)
PHI(523)	missile roll angle, TP (rad)
GAMMAV(527)	vertical flight path angle, TP (rad)
VEL(528)	magnitude of missile velocity (ft/s)
VAT(529)	missile velocity (ft/s)
TF(550)	program termination time (s)
VAH(561)	computed speed of sound (ft/s)
ALAT(576)	latitude of target position (deg)
AZ(578)	azimuth of target position (deg)
DYNP(581)	dynamic pressure (psi)
FLGRJ(606)	constant set to zero indicates run will use ENGINE subroutine and when set to one run will use RAMJET subroutine (simplified ramjet model)
T(932)	actual time (s)
T1(933)	boost engine ignition (s)
T2(934)	commence acceleration command mode (s)
T3(935)	boost engine burn-out/port cover blow-in (s)
T4(936)	start ramjet engine (cruise) (s)
T5(937)	commence heading and altitude guidance control (s)

Symbol	Definition
T6(938)	commence terminal dive (s)
T7(939)	commence terminal guidance - search (s)
T8(940)	commence terminal guidance - track (s)
DMAX(1740)	maximum fin deflection (rad)
CPP(2669)	time between printouts (s)
ROLLO(2901)	initial roll angle (rad)
PITCHO(2902)	initial pitch angle (rad)
YAWO(2903)	initial yaw angle (rad)
STEP(2905)	determines executive program flow after staging
DOC(2909)	defines number of times COMMON will be printed
HMIN(2911)	minimum integration step size
HMAX(2912)	maximum integration step size
DER(1)(2913)	integration step size (s)

ACKNOWLEDGEMENT

This author is grateful to Professor D. L. Collins, NAVAL POSTGRADUATE SCHOOL, for his helpful discussions and suggestions. I also acknowledge NWC CHINA LAKE for their cooperation in obtaining the MOD6DF computer program.

I. INTRODUCTION

Research was undertaken to convert the MOD6DF computer program received from NWC China Lake for use on the IBM-360 computer at the Naval Postgraduate School. Once a functioning program was obtained, initial displacement error versus impact miss distance was investigated for the supersonic tactical missile. Additionally, the midcourse guidance reference altitude was changed to reflect a sea-skimming scenario and this was examined for its effect on the terminal guidance problem. This report not only discusses the aforementioned topics, but also describes the missile mission requirements and the MOD6DF computer program.

II. MISSION REQUIREMENTS

The Supersonic Tactical Missile (STM) mission is divided into six phases (Ref. 1).

* initial conditions
* separation
* boost
* transition
* cruise
* terminal

These divisions are based on the missile aerodynamics. The initial condition phase establishes the starting conditions for each launch. This is done while the missile is still attached to the launch platform. The separation phase starts when the missile is launched. The missile falls for approximately five seconds until the boost engine ignites. This initiates the boost phase which continues until the missile achieves Mach two. As the missile passes through Mach one, plume effects are encountered which the aerodynamics account for. At the end of the boost phase, the port covers blow in. This initiates the transition phase. This phase is very short and allows the debris to be ejected from inlet ports. The cruise phase commences when the ramjet engine ignites. This engine propels the missile until target impact. The terminal phase of the flight begins when the missile is commanded to dive from the cruise altitude. This phase concludes when the flight is terminated at target impact.

The six STM mission phases require four phases of control. These control phases are:

* separation
* midcourse guidance
* terminal dive
* terminal guidance

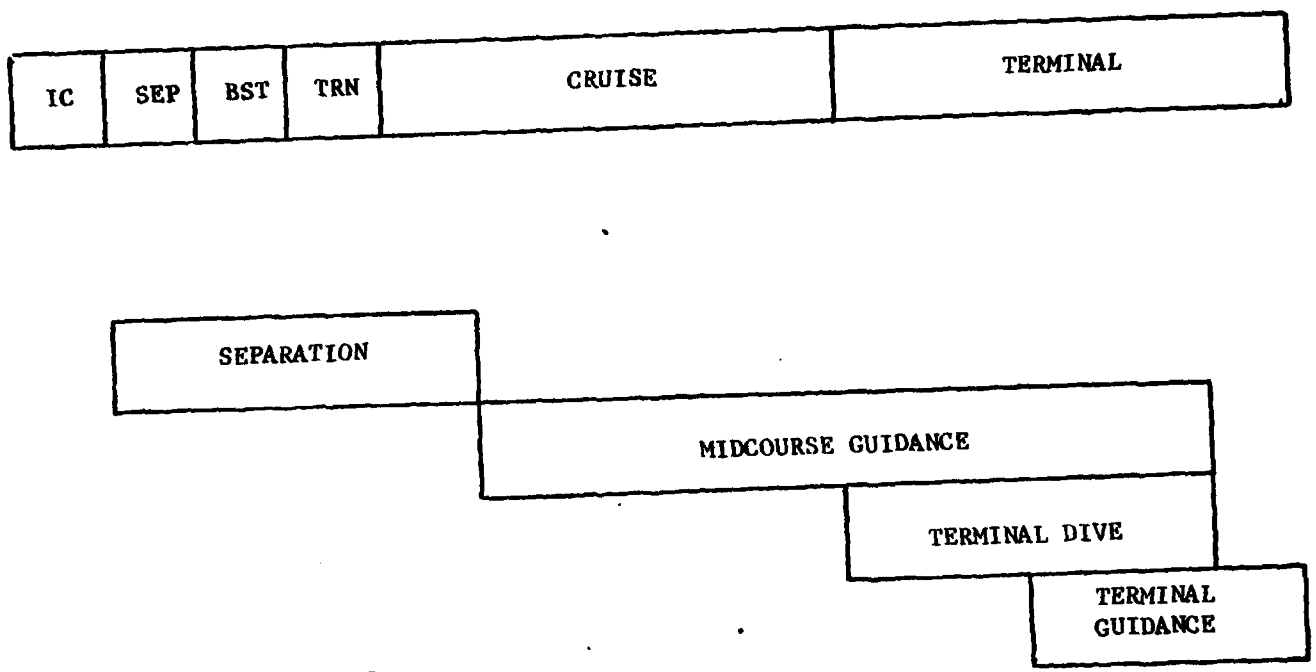
The four Guidance, Navigation, and Control (GN&C) system control phases are directly related to the mission phases. Figure 1 displays this relationship (Ref. 2). To obtain a successful flight, various types of control and guidance functions are required. Figure 2 (Ref. 3) illustrates the guidance control mode sequencing in relation to the mission phases and also indicates the critical missile switching times. The control phases are discussed below, along with the appropriate guidance modes.

A. SEPARATION

The separation phase commences upon launch. The missile is ejected downward from the launch platform. Additionally, the pitch attitude of the missile is commanded down. Since this portion of the flight is unguided, the ejection force and gravity are the only forces acting on the missile. At launch, the missile is required to be in the attitude command mode.

Five seconds into the flight, the missile pitch attitude is commanded up and the boost engine is ignited. The booster continues until the missile attains Mach two, at which time control is shifted from attitude to acceleration control mode. The acceleration control then requires the missile to maintain the normal and lateral accelerations at zero.

The last event to occur in the separation phase is inlet port cover blow-in. The time delay that allows the port covers to clear is a function of altitude.



IC - INITIAL CONDITIONS
SEP- SEPARATION
BST- BOOST
TRN- TRANSITION

CONTROL PHASES/MISSION PHASES
RELATIONSHIP

FIGURE 1

t_0 - LAUNCH: START SIMULATION

t_1 - BOOSTER IGNITION

t_2 - CONTROL SYSTEM MODE CHANGE FROM ATTITUDE TO ACCELERATION CONDITION AT M=2.0

t_3 - PORT COVER BLOW-IN, BOOSTER BURNOUT

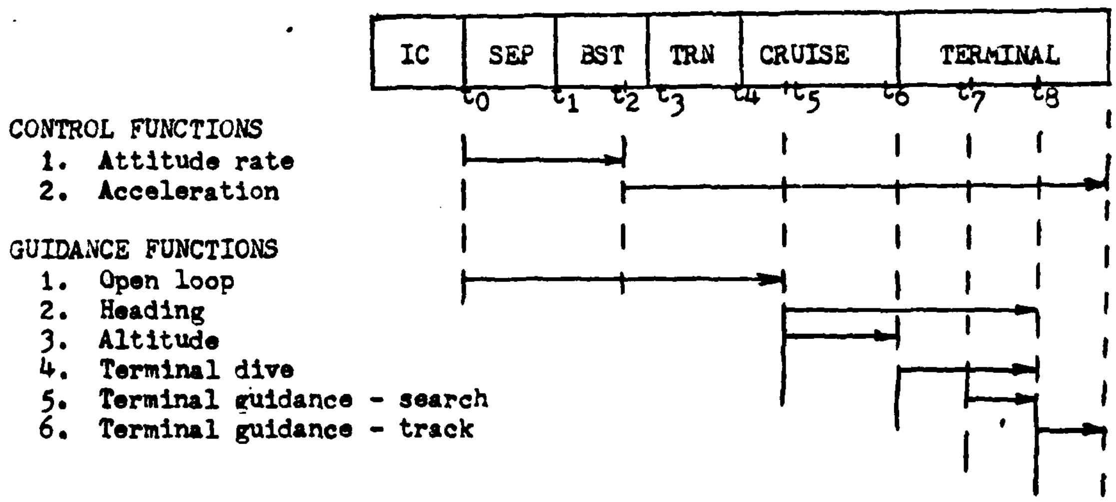
t_4 - RAMJET IGNITION, t_3 + .3 sec

t_5 - ENGAGE FUIDANCE MODES

t_6 - DIVE COMMAND

t_7 - SEEKER SEARCH MODE IS ACTIVATED

t_8 - TERMINAL TRACKING



FLIGHT MODE SEQUENCING

FIGURE 2

B. MIDCOURSE GUIDANCE

Midcourse guidance begins upon completion of the separation phase. The moment the port covers are clear, the phases shift. During this phase two things are required. The GN&C must guide the missile to the established cruise altitude and then guide the missile (in the horizontal plane) to the predetermined target location.

To accomplish these requirements the GN&C system employs both altitude and heading control. The altitude control acts upon the pitch axis to drive the missile to the cruise altitude and then to maintain that altitude until the terminal dive phase commences.

The actual guidance work is performed by the yaw axis. The GN&C system uses the heading control to steer the missile to the target position. To generate the steering commands a guidance law is necessary. The guidance law should minimize the cross-track error and the final lateral acceleration. A minimum cross-track error allows for minimum flight time and minimum displacement error when target search is initiated. Minimizing the final lateral acceleration ensures that the seeker will continuously be pointing toward the target area. To insure accuracy, the guidance law must be able to perform these functions when subjected to disturbances such as wind and thrust misalignment.

C. TERMINAL DIVE

The terminal dive phase commences when the missile is commanded to dive. The actual time that this command is given is a function of the predetermined target coordinates. Upon diving, the missile must accelerate downward along a 60-degree (from horizontal) dive angle to the target position. Once the missile has steadied up in the proper dive aspect, terminal guidance commences.

D. TERMINAL GUIDANCE

During the terminal guidance phase the seeker is required to search, acquire, and track the target. The GN&C system then uses these tracking signals to direct the missile to the target. The seeker uses a preprogramed search pattern to locate the target. When the target has been acquired, the seeker then commences tracking it and also notifies the GN&C system that the target is being tracked.

Target acquisition takes place when the target falls within the instantaneous four-degree beamwidth pattern as it traverses the scan pattern on the earth's surface. The computer program uses this assumption since the exact missile target acquisition mechanism has not yet been defined.

Upon acquisition, the GN&C system closes the seeker tracking loop. The radiometer error signal is defined as the difference between the look vector and the line-of-sight. This signal is used to reposition the antenna gimbals so the look vector and the line-of-sight coincide. Under the closed loop operation the radiometer output is proportional to the line-of-sight rate and this is the signal used to guide the missile to target impact.

III. COMPUTER SIMULATION

The modularized six-degree-of-freedom (MOD6DF) computer program was developed by the Litton Systems, Inc., Guidance and Control Systems Division (Ref. 4) to be used in analyzing missile guidance and control. The program uses a building block approach, where each module corresponds either to a missile subsystem or an environmental system. In its original form, the program is used primarily for terminal guidance of air-to-surface missiles. The Naval Weapons Center, China Lake modified the original program to specifically apply to the Supersonic Tactical Missile (STM). The modification also allows the user the capability of simulating any portion of the missile flight trajectory.

The MOD6DF computer program consists of four main decks and one auxiliary deck. The main decks include the executive programs, operational subroutines, modules, and input data deck. All the subprograms utilized in the integration and the sequencing of the modules are contained in the executive programs. The operational subroutines are used by the user to control the program while it is running. The psysical system and the environmental subroutines are included in the modules. The input data deck contains all the information the executive programs need to execute the desired run. Lastly, the auxiliary deck consists of all those subprograms (subroutines and functions) that are required by the modules.

Once the user starts making simulations, he must concern himself with program sequencing. Proper sequencing is required to ensure a valid run is achieved. Since it is of such importance, sequencing will be discussed as the final part of this section.

A. EXECUTIVE PROGRAMS

The executive programs are the main core of the MOD6DF program. These subroutines have various functions in setting up, sequencing, integrating, and resetting the program. Since these functions are required by any type of analysis done, the user should never have to change any of these subroutines. Reference 5 should be consulted if the user desires more information about the content of any of the subroutines described herein.

1. Zero

Subroutine ZERO is used to set all the COMMON(3415) locations to the value zero. This is done to ensure that no erroneous information is used in the simulation.

2. Oinpt1

Subroutine OINPT1 is the basic input routine for the MOD6DF program. The normal input is from punched cards. However, inputs may also be read from tapes. The basic input cards will be discussed in the section covering the input data deck.

3. Auxi

Subroutine AUXI is used to call the initialization modules for each run. Additionally, it sets up the list of state variables which are used in AMRK.

4. Auxsub

Subroutine AUXSUB is used to call the dynamic modules. In calling the dynamic modules, AUXSUB sets and resets the lists needed for AMRK and the COMMON(3415) storage cells with the most recent values of the state variables and their derivatives.

5. Amrk

Subroutine AMRK is the integration subroutine. It uses a pointwise first order Runge-Kutta method. All the state variables to be integrated must be listed in COMMON(3415) and also must appear in a processing list.

6. Reset

Subroutine RESET is used to reinitialize up to fifty input parameters. This is done prior to the start of any repeated runs. To indicate which parameters are to be reset, the number one is punched in columns 46-60 on the respective type three card.

7. Return Group

The return group is a collection of all the unused modules. Since all the modules and their initialization modules are called by AUXI and AUXSUB, the unused ones must still remain in the deck. These are required to ensure proper linking when the computer attempts to link all the subroutines. All the subroutines in this grouping contain three cards. The three cards are the subroutine title, and a return and an end card.

8. Sub11, Sub12, Sub13

These subroutines are used to call the operational subroutines that are required. They call the routines in the order prescribed by the input data deck. The number at the end of each subroutine title indicates which operational subroutines it can call. For example, SUB11 can call STGE1.

B. OPERATIONAL SUBROUTINES

The operational subroutines provide the user with control of the program while it is running. The order in which the operational subroutines are called is specified by the input data deck. Since these routines assist the user in controlling the simulation, they can be reprogrammed. However, it is advised that they not be changed until the user has become quite familiar with the overall operation and sequencing of the MOD6DF program. For more in depth knowledge of the operational subroutines the user should consult Ref. 6, and the computer listing, which is contained at the end of this report.

1. Inpt1, Inpt2, Inpt3

These subroutines are available for new inputs during the simulated flight. The only one presently used in the program is INPT1. It is utilized to input a namelist file which is used by ENGINE and is described there.

2. Oupt1, Oupt2, Oupt3

These subroutines allow for the print-out of up to fifty different variables during the flight simulation. OUPT1 is not utilized in the deck. OUPT2 is used when the desired output is to be put on tape. OUPT3 is the basic output routine. It prints the desired output on regular computer paper.

3. Stge1, Stge2, Stge3

These subroutines allow for proper staging, run termination, etc.. Presently STGE1 is not being used. STGE2 is being used as the staging initialization subroutine. STGE3 is the primary subroutine of this group. It stages when impact with the earth is made, when the final time, TF(550), is reached, or when LCONV(2672) is set equal to two. All the tolerances for staging are listed in STGE3.

4. <u>Cntr1, Cntr2, Cntr3</u>

These subroutines allow the external dynamic control inputs to the modules. In the MOD6DF program these are not used.

5. <u>Rndm1, Rndm2, Rndm3</u>

These subroutines allow random noise to be added to the state variables generated in the modules. RNDM1 is not used in the program. RNDM2 is used as the initialization subroutine, while RNDM3 provides continous noise values. These subroutines produce correlated noise values for as many modules as required. The noise values remain fixed during each individual integration cycle.

6. Auxa1, Auxa2, Auxa3
Auxb1, Auxb2, Auxb3
<u>Auxc1, Auxc2, Auxc3</u>

These subroutines are auxiliary routines that allow for external input, output, control, etc. of the modules. At the present time, none are utilized in the MOD6DF program.

C. MODULES

The modules are of prime importance to the user since they represent the 'model' of the dynamic system. In general, the model is described by ordinary non-linear time-varying differential equations with both random and deterministic forcing functions. The user must first reduce these equations to an equivalent system of first order equations, which can then be described by each module. Generally speaking, the physical system is so complex that this would be impossible to do. However, due to the modularity of the MOD6DF program, the user can think of each module as a completely independent system described by the equations within that module.

There are thirty-six possible modules divided into five functional

categories. Each group is identified by a letter which pertains to that groups function; A (airframe), C (computers), D (dynamics), G (geophysical), S (sensors). A complete printing of each module is contained in the computer program listing.

1. Airframe

a. Subroutine A1

This is the aerodynamic forces and moments modules. It calculates all the necessary forces and moments in body axes. These values are then used in the computation of the dynamics.

b. Subroutine A2

This is the missile aerodynamic coefficient module. It calculates the required coefficients using the information stored in the BLOCK DATA. Using the timing inputs, this routine computes the coefficients for the different effects. Some of the effects accounted for are; plume, separation, and control surfaces effectiveness. With this done the total coefficients are determined.

c. Subroutine A3

This is the missile propulsion module. The timing inputs are used to determine whether the missile is in free fall, boost, transition, or cruise phase. With this determined the correct engine subroutine (BOOST, RAMJET, ENGINE) can be called. Three variables are calculated using the body axes as the frame of reference. They are the missile thrust is all coordinate directions, the principal moments of inertia, and the missile weight.

d. Subroutine A4

This is the fin actuator module. The four control surfaces commands are calculated as either ideal actuators or as second-order

ones. In addition to control surfaces commands, the rate of change of yaw, and roll are computed.

e. Subroutine A5

This module is part of the return group.

2. Computers

a. Subroutine C1

This is the autopilot module for the STV-G. It uses the cruise engine ignition time to divide the routine into boost/separation and cruise phases. These two phases use different algorithms to calculate the turning moments in pitch/yaw and roll.

b. Subroutine C2

This is the guidance command module. Using the timing inputs, this routine is divided into separation/boost, dive/climb, cruise, terminal dive, and terminal homing sections. Each section uses slightly different algorithms to calculate the guidance commands to maintain the proper flight profile.

c. Subroutines C3 - C10

These modules are part of the return group.

3. Dynamics

a. Subroutine D1

This is the translational dynamics module. It computes the total acceleration in body axes and then converts them to the tangent plane reference. Then, accounting for aerodynamics, thrust, gravity, and coriolis, the velocity and acceleration are calculated.

b. Subroutine D2

This is the rotational dynamics module. With the principal axes as a reference, this subroutine computes the body angular rates and the attitude direction cosines.

c. Subroutines D3 - D5

These modules are part of the return group.

4. <u>Geophysical</u>

a. Subroutine G1

This is the gravitational and coriolis acceleration module. It calculates the gravitational acceleration using one of two fields. The user specifies the field to be used by an input card. To use a flat-earth gravitational field, GZRO(404) must equal 0.0, and to use a spherical gravitational field, GZRO(404) must equal 1.0.

b. Subroutine G2

This module is part of the return group.

c. Subroutine G3

This is the air data module. It computes the velocity, in all three coordinate directions, with respect to the air mass. These values are then resolved into body and stability axes. This module also computes all the properties of air by calling subroutine AIR. These values are stored in their COMMON(3415) locations for use in the other modules.

d. Subroutine G4

This module is part of the return group.

e. Subroutine G5

This is the coordinate conversion module. It takes the missile position, does a coordinate conversion and then it determines the position, velocity, and acceleration in the ECI system.

f. Subroutine G6

This module is part of the return group.

5. **Sensors**

a. Subroutine S1

This is the homing seeker module. It simulates the missile seeker and computes the seeker dynamics and Euler angle rates. Several flags are used to control the seeker sequencing:

(1) FLAGS(335). FLAGS signals the start of the seeker search.

(2) FLGT(317). FLGT signals when the seeker is locked-on.

(3) FLGTS(347). FLGTS signals the end of search.

(4) FLGD(336). FLGD signals when the seeker has detected the target.

(5) FLAGLT. FLAGLT signals when the target is outside the seeker field of vision.

b. Subroutine S2

This is the radiometer module. It takes the target position and the ATIGS target position and converts them from body axes to the seeker axes. Using target position, the module then calculates the azimuth and elevation error signals.

c. Subroutines S3, S4

These modules are part of the return group.

d. Subroutine S5

This is the accelerometers and gyros modules. The user has the option of using ideal or digital accelerometers. To specify the type of accelerometer, the user must include the appropiate data statement in the subroutine. If ideal accelerometers are desired, FLGA must equal 0.0 and for digital accelerometers, FLGA must equal 1.0.

e. Subroutines S6 - S10

These modules are part of the return group.

D. INPUT DATA DECK

The input data deck provides the user with the means of specifying which operational subroutines and modules are to be utilized for the desired run. It also allows the user to set the starting conditions. In general, only the constants and the state variables must be given initial values. All quantities in COMMON not given initial values will be set to zero by ZERO. In addition to the state variables, the upper and lower error bounds must be initialized. There are seven types of input cards, each indication a certain function.

Type	Function
0	read/write tape
1	operational subroutine to be called
2	module to be called
3	numerical input
4	printed output
5	parameter square and sum
6	termination and random noise generator input

A separate card is required for each subroutine, module, input, and output quantity. A sample computer printout of the input data deck is contained in the computer output section.

1. Type 0 Card

Type 0 cards are used to indicate if the type 3 inputs are to be read from or written onto an auxiliary tape. These procedures can be used rather than reading the inputs from a deck of cards. A typical card is defined by punching a zero in column 2. The field that covers columns 5 - 20 is used by the user for any descriptive statements with which he wishes to identify the input. Column 21 -25 contains the right-

justified integer number of the tape transport to be used. The number of the first record to be read is punched in column 31 - 45. The last field is column 46 - 60 which contains the number of records to be read. The last two fields may use either fixed or floating-point notation.

2. Type 1 Card

Type 1 cards are used to specify which operational subroutines are called during the flight simulation. This type card is identified by the number one in column 2. The second field is column 5 - 20 which contains any identifying information. This information is printed out when the data deck is read and allows the user to read exactly which subroutines were called. Column 21 - 25 contains the right-justified integer number which is the subroutine identifying number. The operational subroutine numbers are;

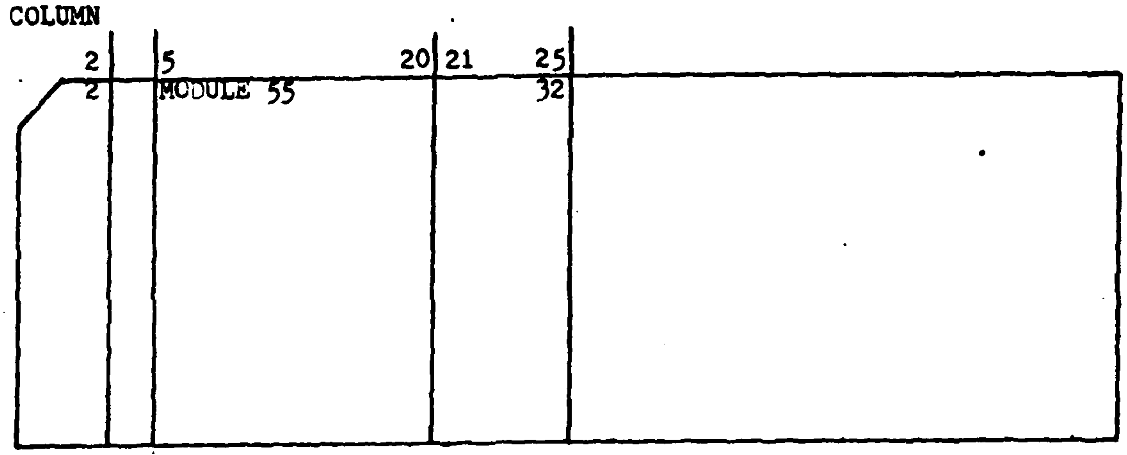
Subroutine	Subroutine Number
INPT1, INPT2, INPT3	2
OUPT1, OUPT2, OUPT3	3
STGE1, STGE2, STGE3	4
CNTR1, CNTR2, CNTR3	5
RNDM1, RNDM2, RNDM3	6
AUXA1, AUXA2, AUXA3	7
AUXB1, AUXB2, AUXB3	8
AUXC1, AUXC2, AUXC3	9

It should be noted that all or any of the subroutines listed under one one number can be called by including only one card. The cards are placed in the data deck in the order in which they will be called. This order or sequencing will be explained further in section F, Sequencing. A typical type 2 card is identified by the number two in

column 2. In general, column 5 - 20 should contain the module title, but any pertinent information is allowed. The module number is punched in column 21 - 25 and it must be right-justified. A listing of the module numbers follows:

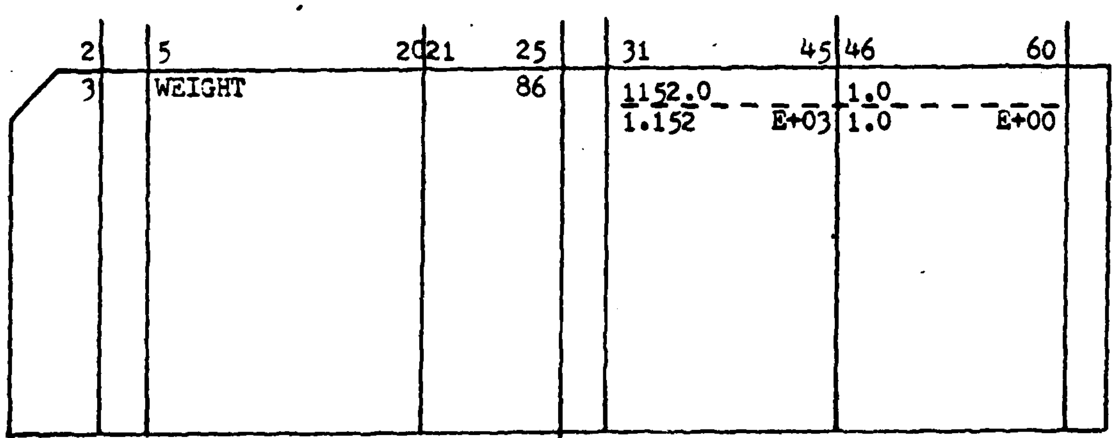
Module	Module Number	Module	Module Number
A1,A1I	2	D4,D4I	20
A2,A2I	3	D5,D5I	21
A3,A3I	4	G1,G1I	22
A4,A4I	5	G2,G2I	23
A5,A5I	6	G3,G3I	24
C1,C1I	7	G4,G4I	25
C2,C2I	8	G5,G5I	26
C3,C3I	9	G6,G6I	27
C4,C4I	10	S1,S1I	28
C5,C5I	11	S2,S2I	29
C6,C6I	12	S3,S3I	30
C7,C7I	13	S4,S4I	31
C8,C8I	14	S5,S5I	32
C9,C9I	15	S6,S6I	33
C10,C10I	16	S7,S7I	34
D1,D1I	17	S8,S8I	35
D2,D2I	18	S9,S9I	36
D3,D3I	19	S10,S10I	37

Note that either the module, the initialization module, or both may be called by including only one card in the deck. A sample of a typical type 2 card is shown in Figure 3 (Ref. 7).



TYPICAL TYPE 2 CARD

FIGURE 3



TYPICAL TYPE 3 CARD

FIGURE 4

4. Type 3 Card

Type 3 cards are used to set any COMMON(3415) location to any value other than zero. In general, four items must be initialized. The state variable initial values and any constants used in the flight simulation are the most obvious. Additionally, there are some constants associated with the executive programs and operational subroutines and the state variable upper and lower bounds which must be initialized. As with all cards, column 2 defines the type card and it must contain a three. Column 5 - 20 holds the statement describing the input. The user should be specific here since it will save him having to remember every COMMON(3415) location. The only other means of input identification is by column 21 - 25. These columns contain the right-justified COMMON location of the input. Columns 31 - 45 hold the actual numerical value of the input. The last field, column 46 - 60, contains the reset flag. If the reset flag equals one, the COMMON(3415) location and the numerical value are placed in the reset list. This list may contain up to fifty different variables. This, in the case of multiple runs, allows the variables to be reset to its initial value prior to each run without additional input cards. The sample card in Figure 4 (Ref. 8) shows that either fixed or floating-point notation may be used to input the numerical value and the reset flag. These cards need not be inputed in any specific order, but for ease of checkout, it is advised to place them sequencially by their COMMON location.

5. Type 4 Card

The MOD6DF program can printout a maximum of fifty variables for each simulation. Type 4 cards are used to specify which variables are to be printed. Column 2 must contain the number four to indicate a type

4 card. When the results are printed out headings are included. These headings are designated in column 9 - 20. The exact alphanumeric title punched will be printed at the top of each page, this need not be the fortran symbol used within the program. The COMMON(3415) location of the output variable is contained in column 21 - 25 and must be right-justified.

6. Type 5 Card

Type 5 cards are used to indicate which variables are to be root-mean-squared. These cards are similar to type 1 and type 2 cards. Column 2 must contain the number five. Any pertinent information about the variable to be operated on is punched in column 5 - 20. The COMMON (3415) location of the variable must be right-justified in columns 21 - 25. The last field, column 31 - 45, indicates whether the root-mean-square operation will occur along the trajectory or at the end.

7. Type 6 Card

The type 6 card has two purposes. Its first function is trivial, but required. The number six is typed in column 2 and the rest is blank. This indicates to the computer program that there are no more input cards. Its second function involves random inputs. This card is used to indicate the number of random process (noise) generator cards that are to be read before the input process is terminated. Column 2 has the same information as before. Any pertinent information is contained in column 5 - 20. Columns 21 - 25 must be right-justified and they contain the number of random generator cards to be read.

E. AUXILIARY DECK

The auxiliary deck is a collection of subroutines and functions required by the modules. These routines are general in nature since

they can be called by several modules. They are used to calculate such things as the properties of air, engine performance, various ratios, and to locate values in the many data tables. A brief discussion of the most common subroutines is presented.

1. Boost

This subroutine calculates the thrust coefficient (CT) and the fuel flow rate (FF) during the boost phase. These values are returned to module A3 and used to calculate the missile thrust in the body axes and the missile weight.

2. Ramjet

This subroutine is used during the midcourse, cruise, phase of the missile flight. It uses a simplified ramjet model to calculate the thrust coefficient and fuel flow rate. This routine is not automatically called. The user must designate that he wishes to use it by inputing a type 3 card setting FLGRJ(606) equal to one. This then sets up the proper stepping in module A3.

3. Engine

This is the primary routine during the cruise phase. It is one of many routines within the NWC air-breathing propulsion package. This entire package is utilized to calculate the engine parameters from inlet to exhaust. Again, thrust coefficient and fuel flow rate are eventually computed and returned to module A3. The user does not need to supply any special input cards to use this subroutine. If no initial value is inputed for FLGRJ(606) it is automatically set to zero, which indicates this routine is to be used.

4. Air

All the properties associated with the air are calculated in

this subroutine. This includes the computation of the speed of sound and the dynamic pressure. Since the missile does vary in altitude, the routine takes this into account as well as the latitude. Once these values are calculated, they are returned to module G3.

5. Block Data

This routine contains data tables. These tables cover parameters from aerodynamic coefficients to thermal properties. The total data package covers specific bands within the missile operating envelope. Data is stored in matrices which includes one, two, and three dimensional ones. These data tables are readily available and there are routines designed to retrieve this information quickly.

6. Serch

This subroutine, along with several functions, is used to retrieve information from BLOCK DATA. If the present operating point of the missile is not within one of the bands of information, then the tables are interpolated. The functions THREDL, STDLI, STDLIA, and TAB do the interpolation of the tables. Since the tables are of various lengths, these functions are very general.

F. SEQUENCING

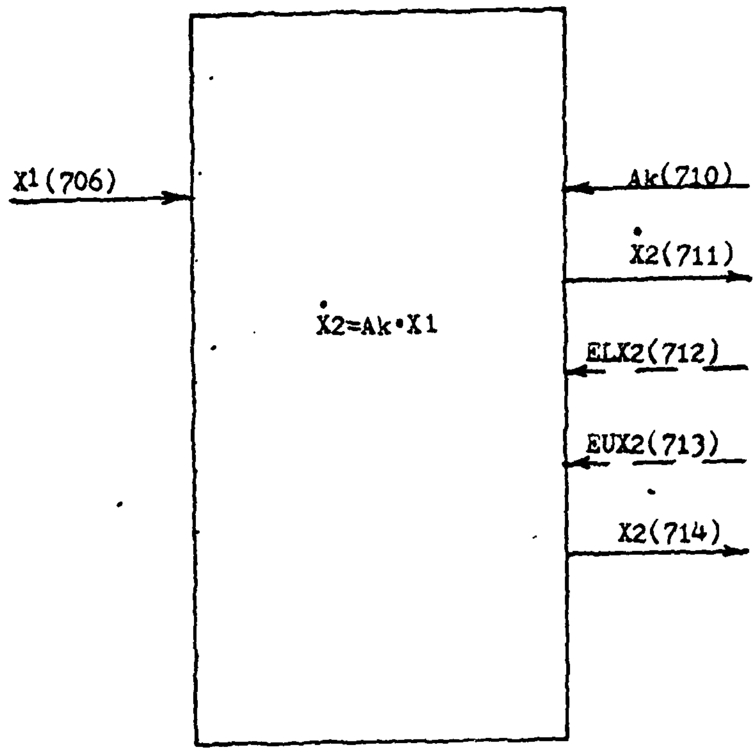
Sequencing is very important in the running of the MOD6DF program. Care must be taken to ensure that the modules are processed in the correct order at each step. This is essential to eliminate the use of obsolete values from the last cycle. An exception to this problem is the state variables. These are updated simultaneously by the integration algorithm. Any module is capable of using the most current value of these, no matter what the order of processing.

To help remedy this problem, module diagrams were devised. Module

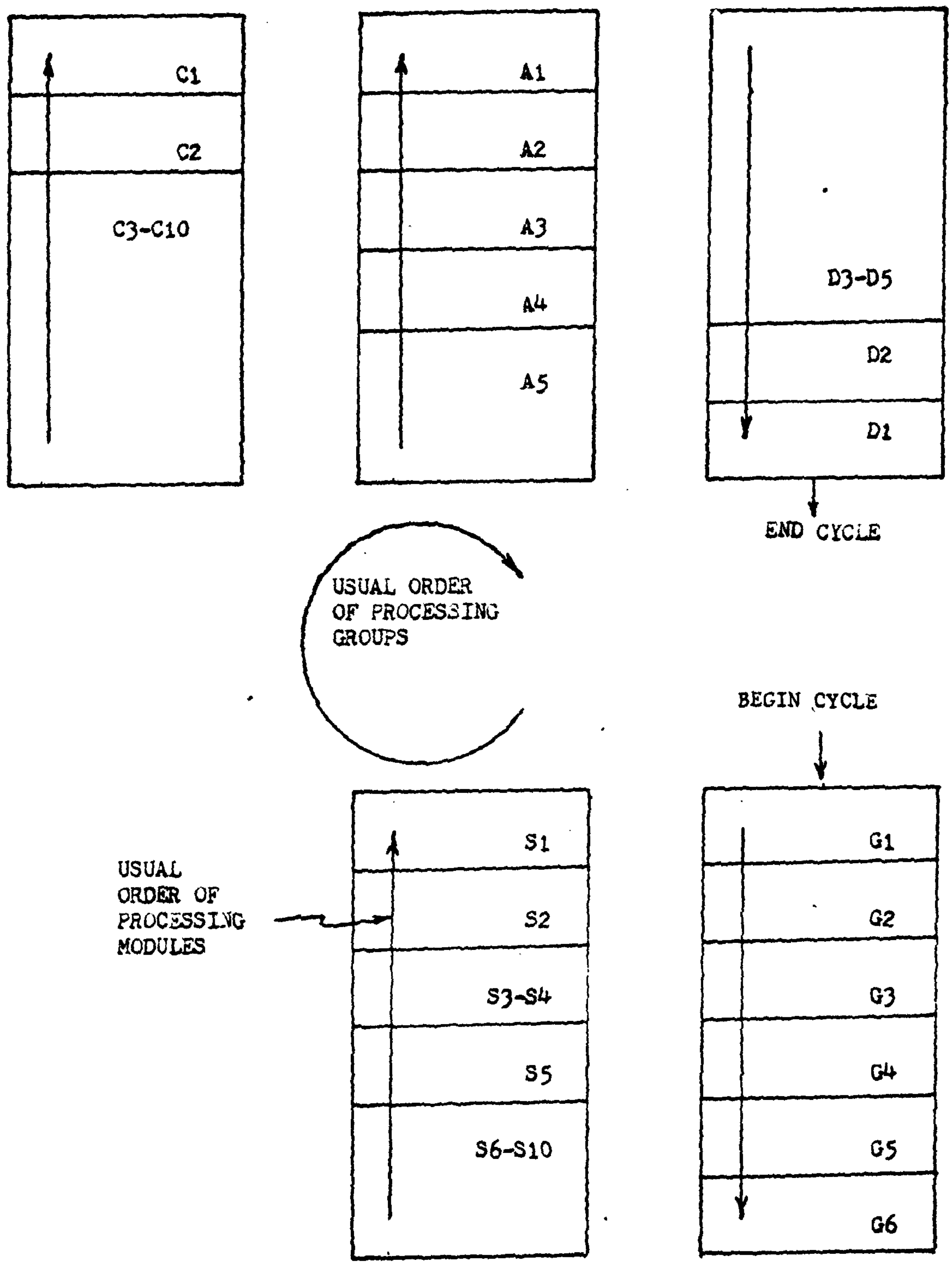
diagrams aid the user in maintaining the proper flow of variables into and out of the modules. To design a module diagram start with a box. This box will contain all the equations for a specific module. The standard procedure for showing inputs and outputs is to use arrows pointing in or out from either the left or the right side. The example in Figure 5 (Ref. 9) shows this technique. The arrows pointing in from the left indicate variable inputs from other modules. Arrows pointing out to the right indicate output going to either other modules or program output. The last set of arrows points in from the right. These show the constants brought in directly or indirectly from the initialization module. Since each arrow represents a variable, they must be defined. The usual means of labeling the arrows is to use the variable fortran symbol and in parenthesis its COMMON(3415) location.

Each variable usually has only one COMMON location associated with it. In the case of 'state' variables this is not true. State variables are defined by four consecutive COMMON locations. The first is for the derivative of the variable. The second and the third locations hold the lower and upper bounds, respectively, of the integration error. The last one contains the variable itself.

Once all the required module diagrams have been completed, they can be combined to get the overall processing order. The usual processing order, shown in Figure 6 (Ref. 10) is to start with the geophysical group, then proceed to sensors, computers, airframe, and finally dynamics. Within each group is a usual processing order and this too is shown in Figure 6. This process for determining the program sequence will eliminate the use of any obsolete values in the computer run.



MODULE DIAGRAM
FIGURE 5



PROCESSING SEQUENCE

FIGURE 6

IV. COMPUTER PROGRAM CHECKOUT

The basis for the research was the MOD6DF computer program from NWC China Lake. The total package received from them consisted of a listing of the program and an uninterpreted deck of cards. The initial step was to input the cards in small groups into the computer and then to examine the source listing. This listing revealed that the original deck was punched in BCD. This fact was easy to determine since several characters were changed (Ref. 11). The library routine NEWDEK was used to translate the cards from BCD to EBCDID.

Once the translation was completed, an attempt was made to compile the new deck. This produced an output which contained many syntax errors. These errors were divided into two major groups. One effected the use of quotation marks in FORMAT and comment statements. The other one, the more difficult, effected the DATA statements in the BLOCK DATA subroutine.

The problem with the DATA statements was due primarily to the difference in the compilers used. The compiler at NWC was much newer and allowed for the use of more sophisticated inputs. The compiler at NPS only allows a data set to start with the first element. This required the rewriting of many data groups. To complicate these revisions, a limit of nineteen continuation cards is also imposed. These restrictions demanded not only the rewriting of many data sets, but also the formation of two new ones.

With the corrections finally completed, the computer would then compile the program. The next step was to link all the subroutines together. The first attempt was unsuccessful. Inadvertently, the subroutine INTR20 had been omitted from the original deck. Using the program listing, the contents of INTR20 were typed and included in the

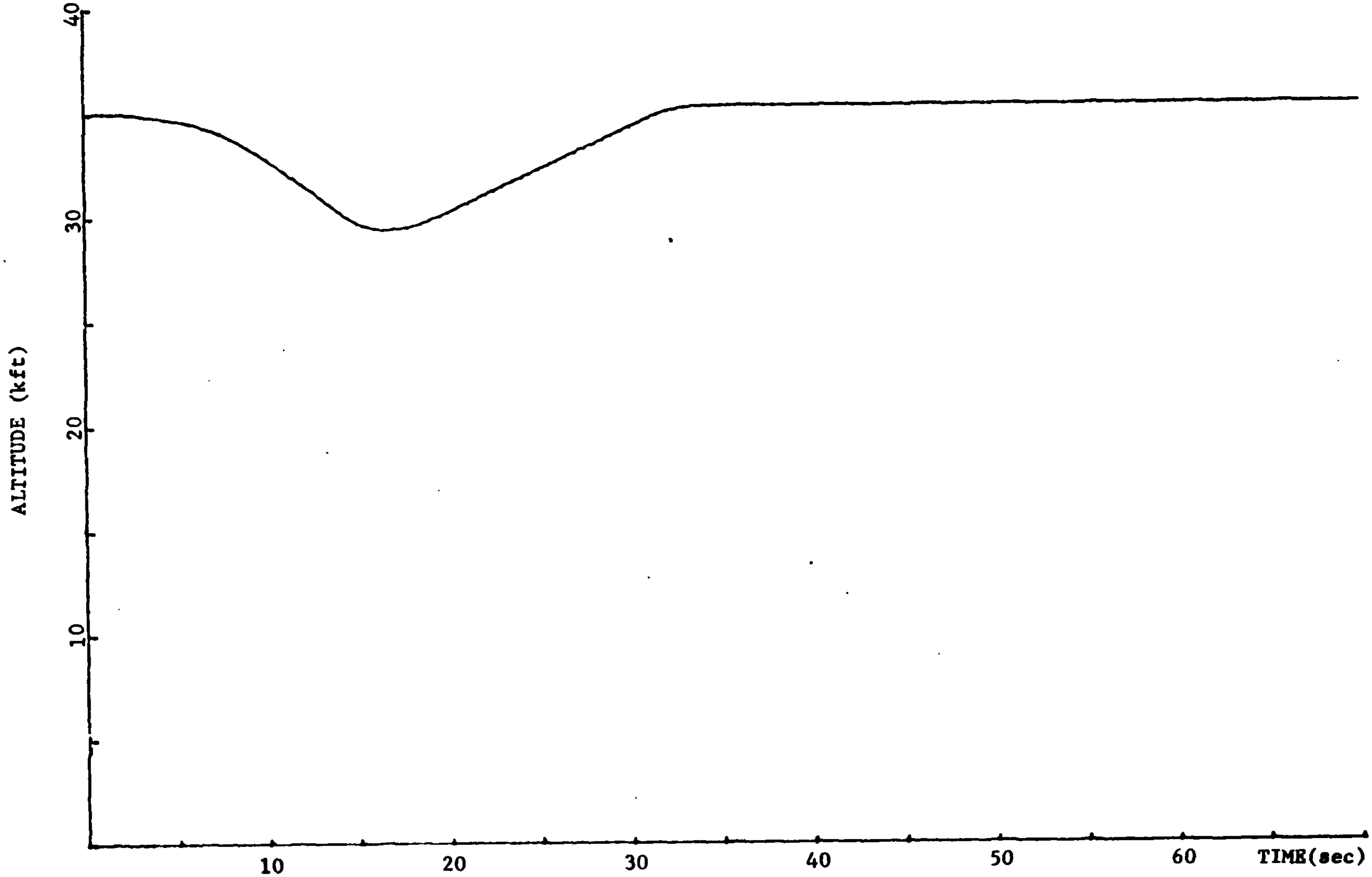
main deck. With this addition the program would now compile and link.

Now that the program would compile and link, attention was turned to getting a good simulation. To facilitate this process, two sample outputs were obtained from NWC. These outputs cover one missile flight which is broken down into a midcourse guidance and a terminal guidance s imulation. Hereafter these will be referred to as midcourse baseline and terminal baseline, respectively. Using the initial conditions from the baseline models, it was hoped that the outputs could be duplicated.

A. MIDCOURSE GUIDANCE FLIGHT

The initial run, using the midcourse baseline inputs, revealed an overflow problem with the dynamic pressure (QD(508)). Us ing the traceback procedures outlined in the Users Manual from the W. R. Church Computer Center, the problem was confined to subroutine AIR. The problem turned out to be a translation error. The symbol PØ (Ø - zero) had translated to PØ in one place and PO in another. This error caused the program to use a value left in that memory location from a previous run to calculate the dynamic pressure. With this problem remedied, the program could progress a little farther. The next stumbling block appeared as a divide check. These errors were resolved by introducing patches that would bypass a statement that tried to divide by zero. Once bypassed, that quantity would be set equal to zero. This was the normal procedure of the computer, but it would stop the run after ten such errors. Having corrected all these errors, output was obtained which covered the desired seventy seconds of flight.

When the output was examined a major switching problem within subroutine A3 (missile propulsion module) was found. The midcourse baseline utilized a simplified ramjet model, but the output was not.



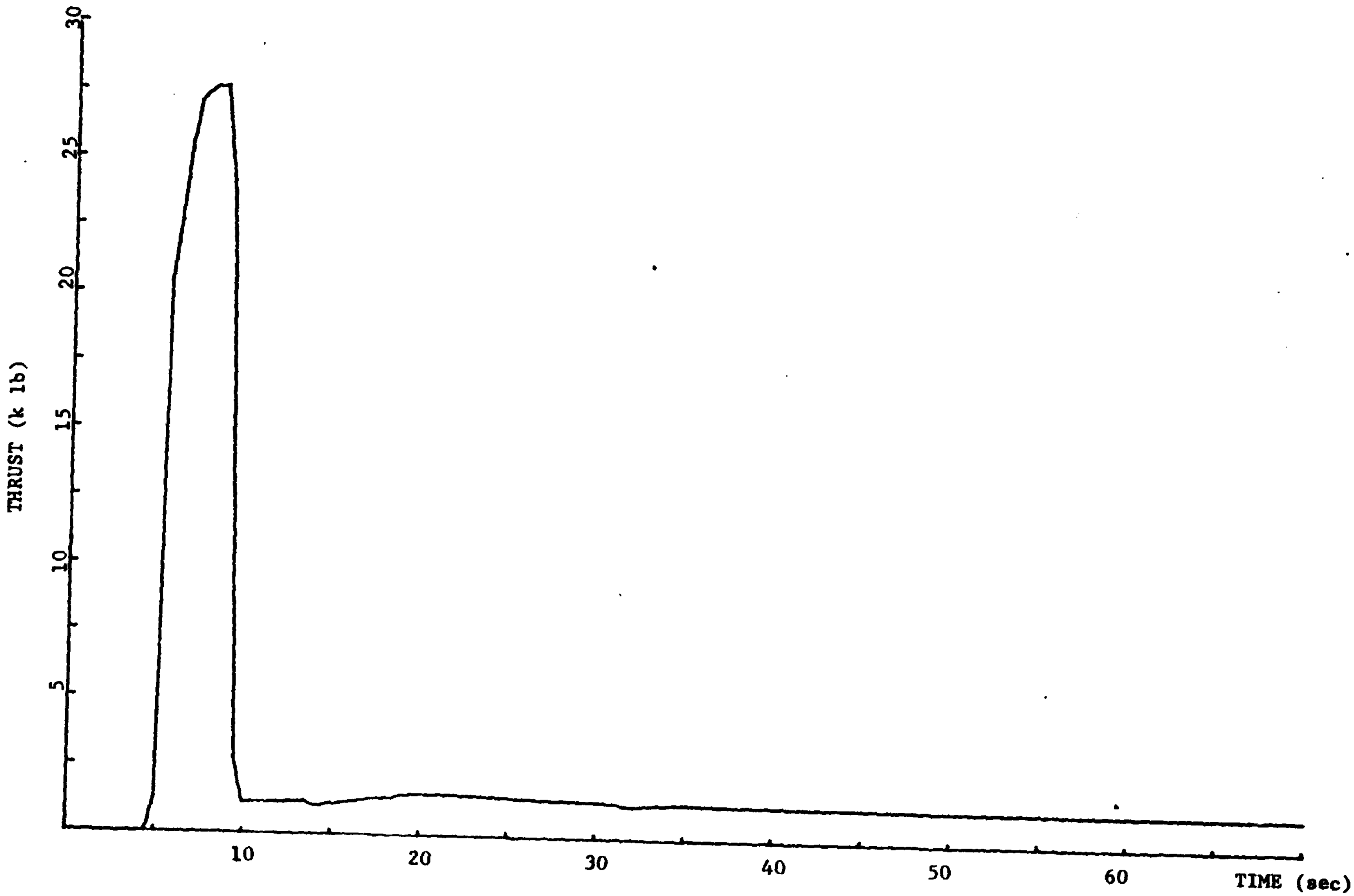
ALTITUDE VS. TIME (MIDCOURSE)

FIGURE 7

Rather than use RAMJET, the output disclosed that ENGINE had been used. The problem was in the logic statement, FLGRJ 0.0. The program was not making the desired switch to the simplified ramjet model. To remedy the problem, the data card FLGRJ(606) 1.0, was added to the input data deck. Another problem was discovered which occured between 14.0 and 14.5 seconds. During that time period the missile angle of attack (ALPHA (330)) exceeded ten degrees. When ALPHA exceeds this angle the engine is turned off. With the engine off, no thrust is produced causing the forward velocity (VXTP(286)) to decrease. This limitation was removed from the program. After eliminating these problems a flight trajectory, Figure 7, was obtained which closely resembled the output of the mid-course baseline. A random sampling of the outputed variables were compared for exactness. The differences noted were due primarily to computer round-off error.

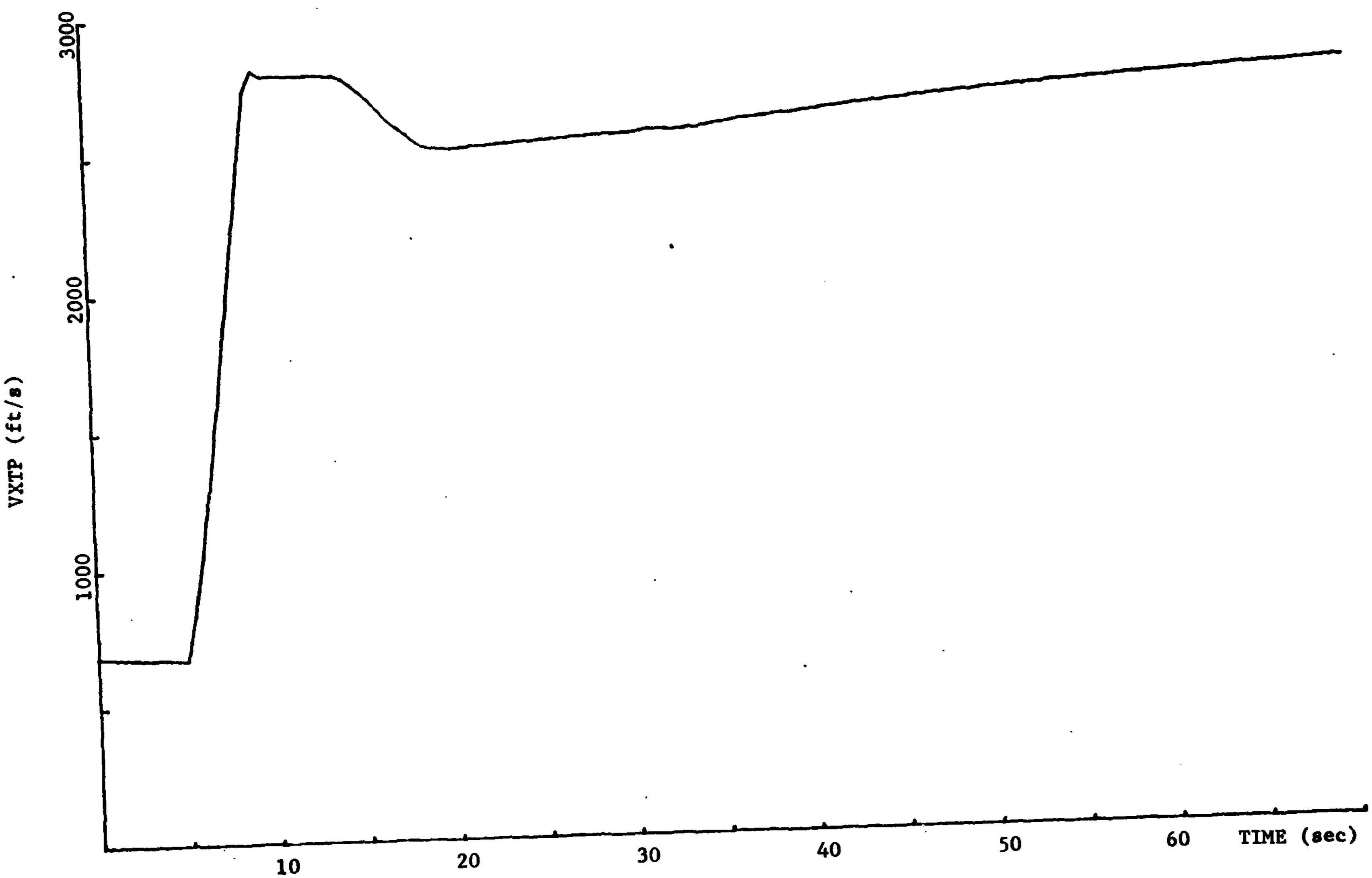
To further verify the accuracy of the two simulations, three parameter (TXBA(073), VXTP(286),and YTP(298))were chosen as representative values for the runs. To obtain a feel for the run, these parameters are plotted in Figures 8, 9, 10. Additionally, random time samples are tabulated in Tables I, II, III.

The thrust (TXBA(073)) plot, Figure 8, can easily be divided into four mission phases. During the separation phase (0.0 - 4.5 seconds) the thrust is zero. This is expected since neither of the engines have ignited. This is followed by a rapid increase in the thrust. The maximum thrust, 27800 lbs, happens during the boost phase (4.5 - 9.5 seconds). At 9.5 seconds the boost motor stops and the transition phase occurs for the next 0.3 seconds. During this time the thrust decreases rapidly since neither engine is on. Once the engine port covers are



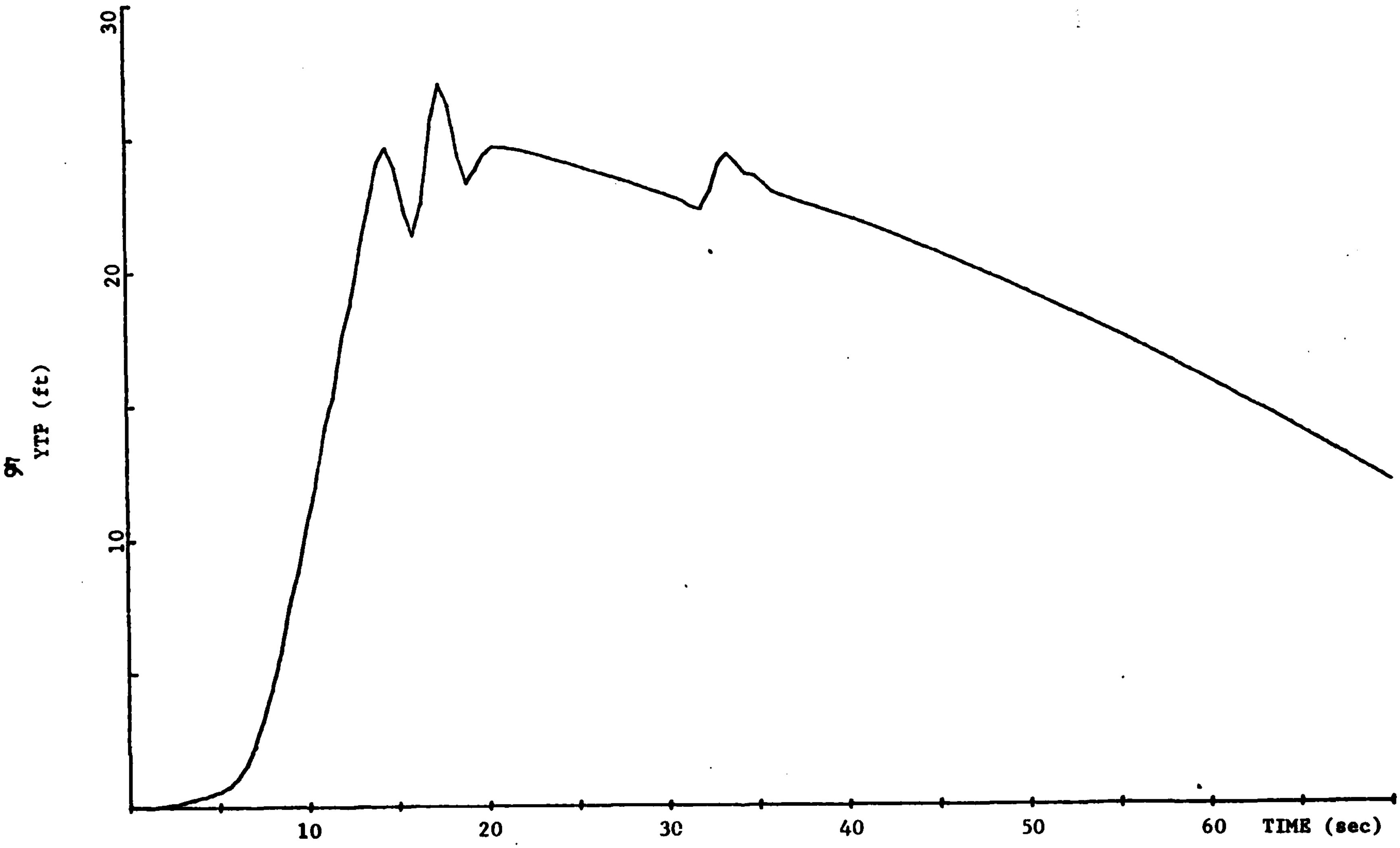
THRUST VS. TIME (MIDCOURSE)

FIGURE 8



VXTP VS. TIME (MIDCOURSE)

FIGURE 9



YTP VS. TIME (MIDCOURSE)

FIGURE 10

clear, the ramjet sustainer ignites. This action initiates the midcourse phase (9.8 - 70 seconds) and the thrust remains relatively constant at 1200 lbs.

The missile forward velocity (VXTP(286)) can be directly related to the thrust. During separation no thrust is produced, therefore the velocity is zero. As the missile is boosted to Mach two, the velocity increases and approaches its maximum value (2800 ft/s) just prior to boost engine shut-down. As the sustainer engine port covers clear, the velocity decreases slightly. Once the ramjet engine ignites the velocity profile is displayed in Figure 9 and it closely resembles that for the midcourse baseline.

The last parameter, y-displacement (YTP(298)), was included to demonstrate the accuracy of the guidance system Figure 10 shows that prior to attaining the cruise altitude, the missile wanders off track. Once the guidance system is activated, it makes the necessary corrections to return the missile to the planned flight path. Table III shows that the y-displacement corresponds to the baseline case and at the conclusion of the run the off-track error is down to 12.03 feet.

All the information obtained from the new run was checked against the midcourse baseline. The results showed that the two simulations are within acceptable limits. With this milestone completed, the program checkout could proceed to the terminal guidance flight.

B. TERMINAL GUIDANCE FLIGHT

To checkout the terminal guidance portion of the MOD6DF program, the initial conditions from the terminal baseline were used. This required changing about a dozen input cards in the midcourse input data deck. The first run attempted turned out successful. This was due to the

TIME	0.0	7.5	10.0	20.0	25.5	30.0	40.0	50.0	60.0	70.0
BASELINE	0.0	27395	1140.6	1403.0	1377.4	1281.0	1211.3	1178.0	1148.1	1124.8
NEW	0.0	27391	1138.9	1514.3	1418.4	1314.3	1249.9	1207.0	1175.5	1149.7

TXBA (073) COMPARISON (MIDCOURSE)

TABLE I

TIME	0.0	7.5	10.0	20.0	25.0	30.0	40.0	50.0	60.0	70.0
BASELINE	681.99	1858.3	2790.2	2606.7	2604.9	2613.4	2652.4	2705.0	2743.8	2771.5
NEW	681.99	1851.7	2786.3	2506.4	2525.9	2544.5	2605.0	2665.5	2710.8	2743.9

VXTP (286) COMPARISON (MIDCOURSE)

TABLE II

TIME	0.0	7.5	10.0	20.0	25.0	30.0	40.0	50.0	60.0	70.0
BASELINE	-.5E-5	3.46	11.37	27.21	25.92	24.34	20.45	16.93	13.24	9.39
NEW	-.5E-5	3.19	10.68	24.43	23.93	22.82	21.96	19.14	15.81	12.03

YTP (298) COMPARISON (MIDCOURSE)

TABLE III

TIME	0	5.0	10.0	12.6	15.0	17.6	20.0	22.6	25.0	TF
BASELINE	1141.9	1131.0	1077.8	1170.9	1309.4	1501.1	1730.0	2245.3	2565.8	2691.3
NEW	1141.6	1134.8	1099.4	1212.8	1359.3	1561.7	1800.0	2305.9	2584.3	3273.6

TXBA (073) COMPARISON (TERMINAL)

TABLE IV

TIME	0.0	5.0	10.0	12.6	15.0	17.6	20.0	22.6	25.0	TF
BASELINE	2750.0	2764.2	2654.6	2471.4	2225.7	1854.5	1391.4	1376.9	1413.1	1425.05
NEW	2750.0	2761.5	2631.5	2438.7	2192.1	1828.2	1389.3	1366.7	1406.1	1465.2

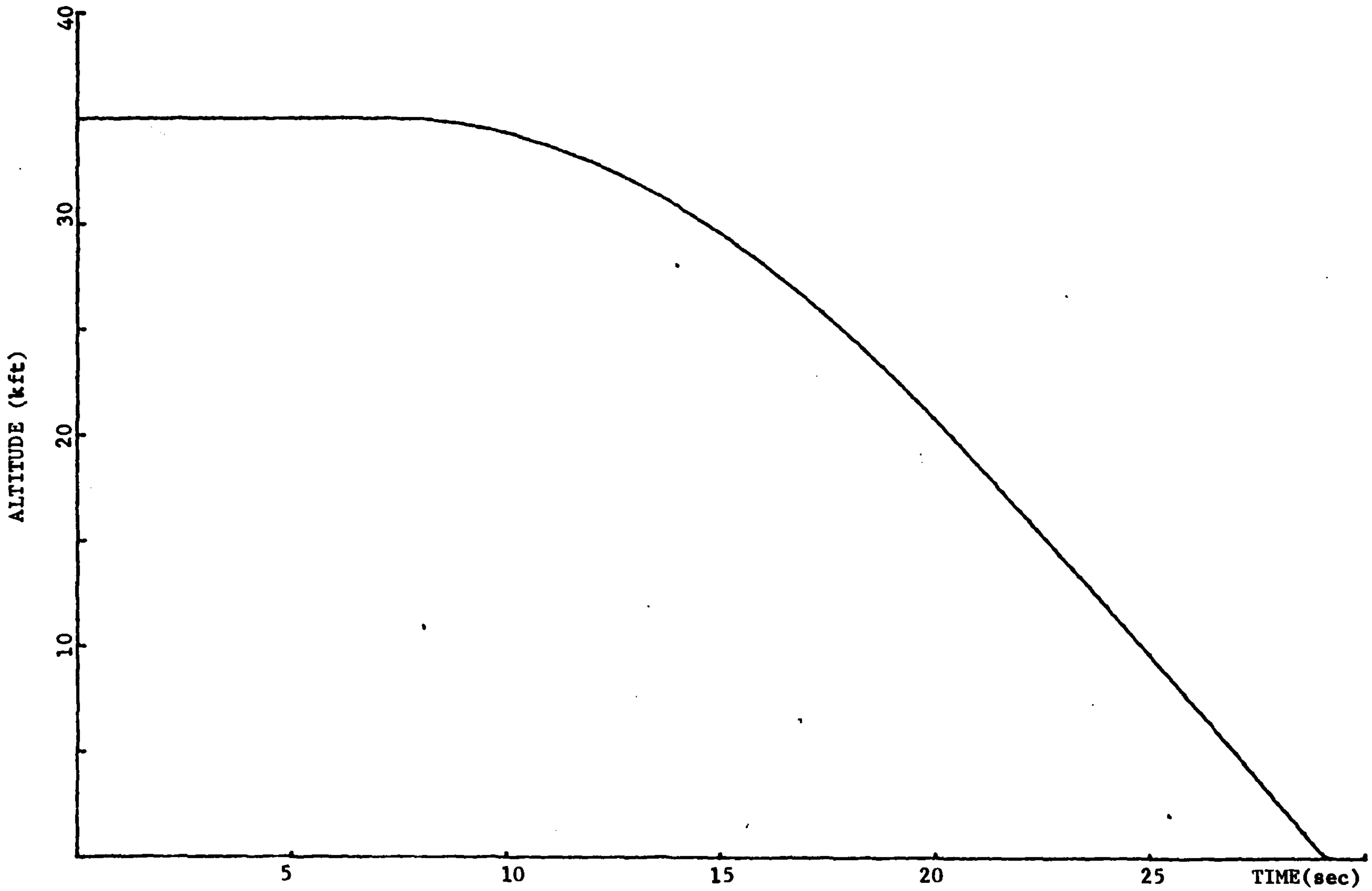
VXTP (286) COMPARISON (TERMINAL)

TABLE V

TIME	0.0	5.0	10.0	12.6	15.0	17.6	20.0	22.6	25.0	TF
BASELINE	0.0	0.0	-.4E-6	-.5E-6	.7E-6	.9E-6	.1E-5	.2E-6	.1E-6	.1E-6
NEW	0.0	-.0074	.019	-1.38	-.095	-.868	-1.77	.071	-.151	-.05

YTP (298) COMPARISON (TERMINAL)

TAVLE VI



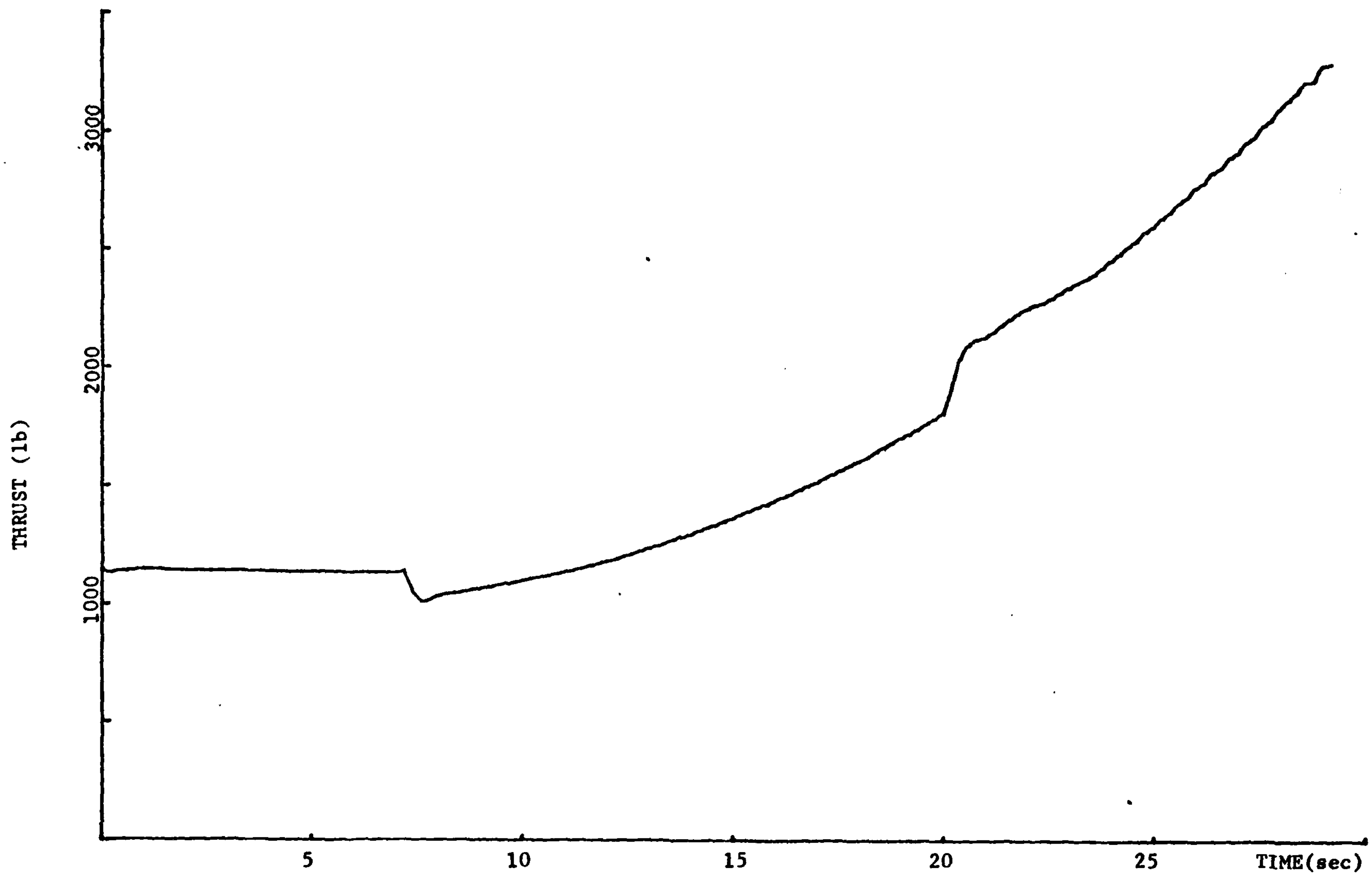
ALTITUDE VS. TIME (TERMINAL)

FIGURE 11

elimination of all the errors during the midcourse checkout. The resulting flight trajectory, Figure 11, is a very good illustration of the desired profile. To check the accuracy of the simulation, the same three parameters were chosen. Tables IV and V show that this run is almost identical to the terminal baseline. The difference is primarily due to computer round-off error. Figures 12 and 13 show the overall flight performance of the thrust and forward velocity, respectively.

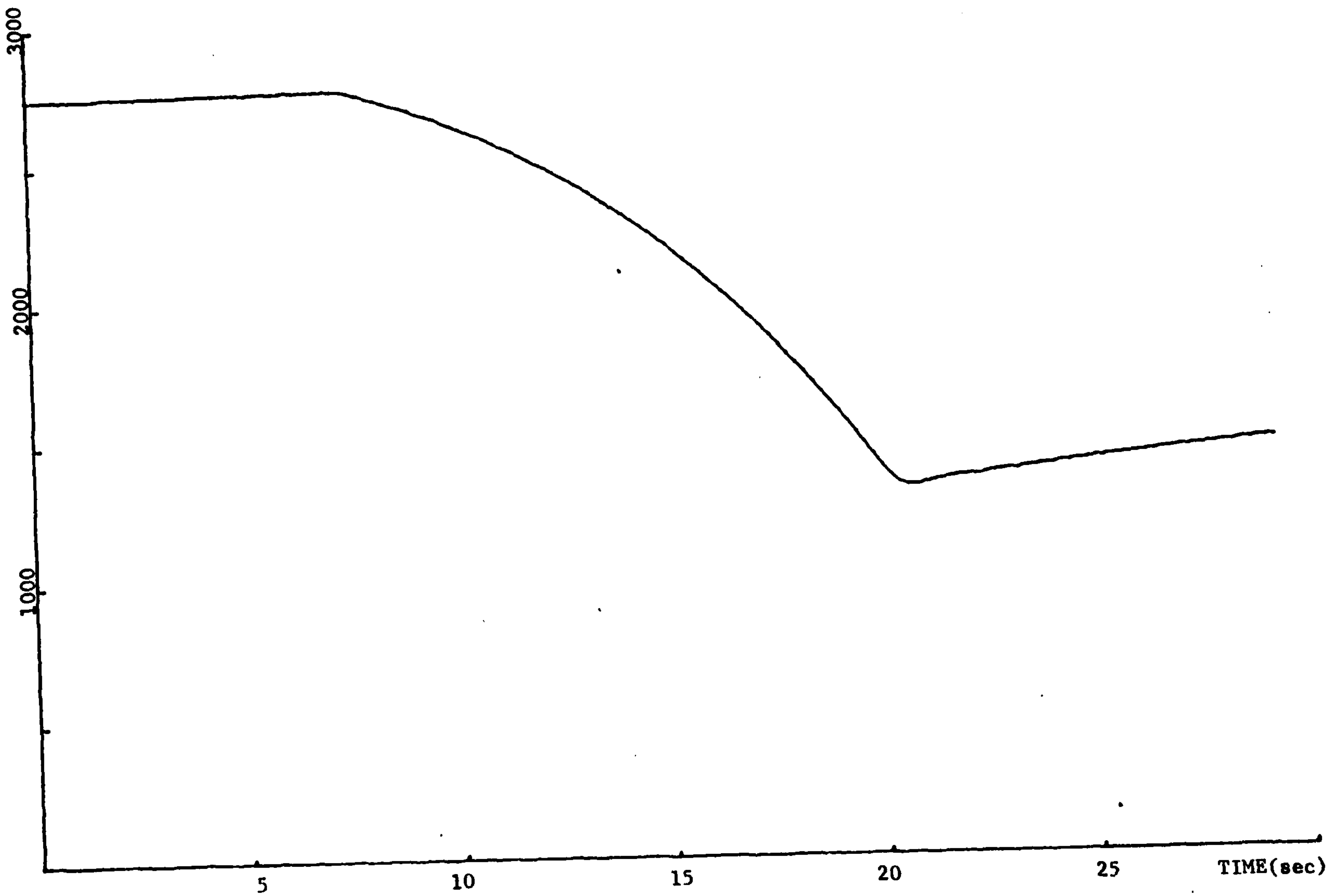
For this simulation, the y-displacement (YTP(298)) does not compare favorably with the terminal baseline. Figure 14 shows that between ten and twenty seconds in the flight, YTP develops rapid oscillations. By examining Table VI, it is noted that during this interval the two simulations differ the most. No reason for this discrepancy could be found. These oscillations do effect the flight by increasing the time of flight from 25.825 to 29.09 seconds and by increasing the miss distance. The imposed time constraints force this problem to be overlooked and to direct attention to the accuracy of the impact. Since the impact is within 0.5 feet of the target location, it is concluded that the terminal guidance simulation works properly.

With both segments of the MOD6DF computer program working correctly, thoughts could now turn towards the actual experimental runs. The results of these runs are discussed in the next section.



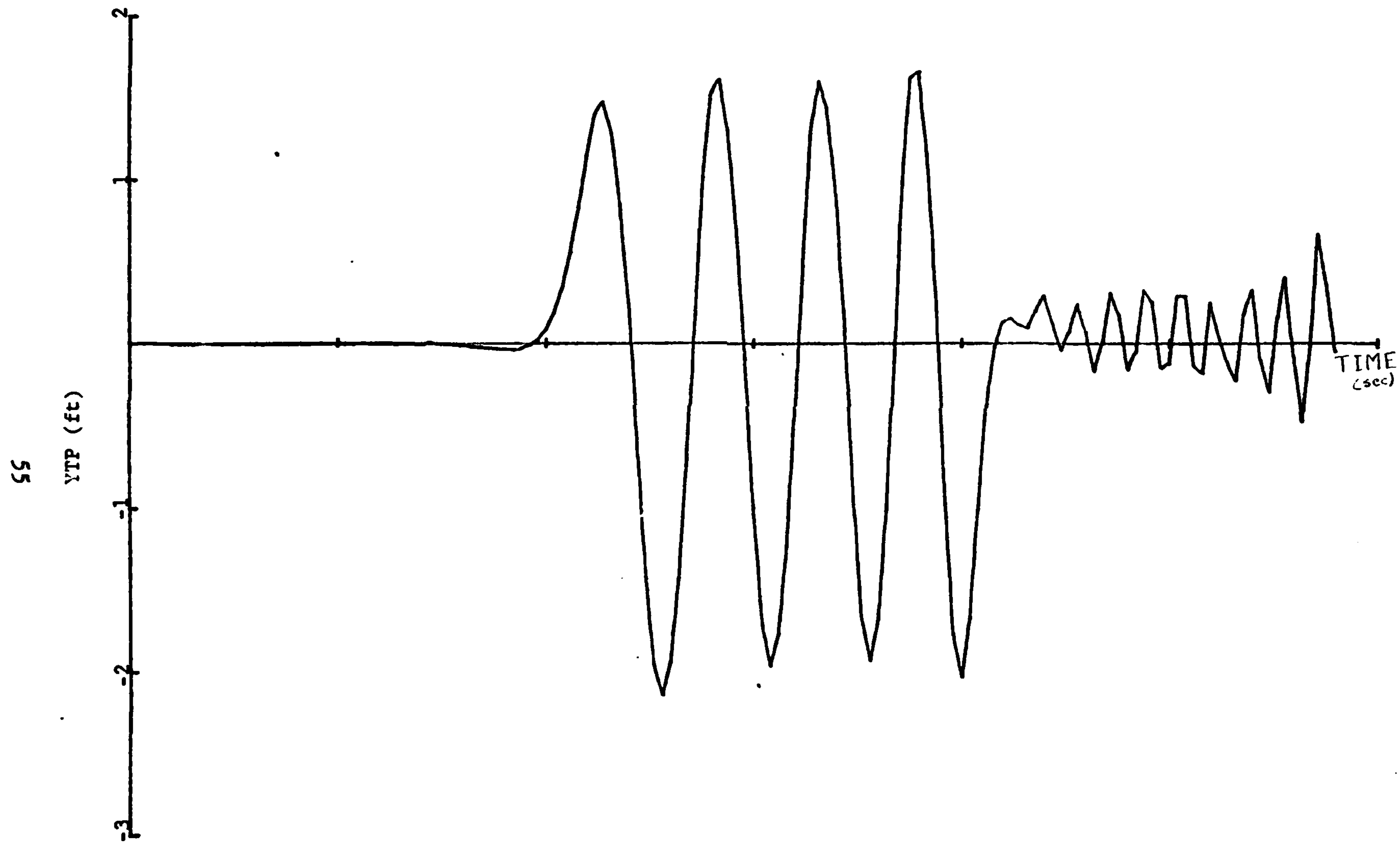
THRUST VS. TIME (TERMINAL)

FIGURE 12



VXTP VS. TIME (TERMINAL)

FIGURE 13



YTP VS. TIME (TERMINAL)

FIGURE 14

V. TERMINAL FLIGHT DISTURBANCES

With the checkout of the MOD6DF computer program completed, many ideas were discussed concerning alterations to the simulations. The three modifications decided upon were:

* examine target miss distance when changes were made to the initial x-y-z conditions,

* examine the terminal flight profile, range, and miss distance when the cruise altitude was reduced to approximate a sea skimming mode,

* examine target miss distance when random noise is applied to the missile homing seeker.

To understand the changes and results, one must be familiar with the frame of reference. The origin of the reference frame travels from the launch platform to the target location at the cruise altitude. This is called the tangent plane reference system. Displacements in the x-direction (XTP(290)) are measured from the launch platform in the direction of the target. Y-displacements (YTP(298)) are measured left or right of the ideal flight path, in the tangent plane. Any vertical displacement (ZTP(306)) is measured normal to the ideal flight path, with down being the positive direction. Using this frame of reference the information in Table VII is easier to understand.

The first modifications demonstrate the effect of changing YTP. YTP was increased until a result was reached that was unsatisfactory. From the results in Table VIII, the maximum value of YTP was determined to be 1800 feet. This conclusion corresponds with the results in reference 13. Since the missile is symmetric, it was also concluded that moving YTP either right or left would give the same results.

The fifth run simulated a drop in the missile's altitude. ZTP was

RUN	XTP	YTP	ZTP
BASELINE	1735552.5	0.0	0.0
1	173552.5	1000.0	0.0
2	173552.5	1500.0	0.0
3	173552.5	1800.0	0.0
4	173552.5	2000.0	0.0
5	173552.5	0.0	2000.0

INITIAL CONDITIONS

TABLE VII

RUN	XTP	YTP	ZTP	IMPACT TIME
BASELINE	234999.56	-.0506363	34999.34	28.825
1	234999.81	.1259258	34999.836	29.108
2	235000.19	.11479032	35000.32	29.124
3	235000.25	.15263242	35000.391	29.134
4	269802.0	2000.0	19705.00	35.0
5	234999.63	.45691	35000.426	30.365

IMPACT RESULTS

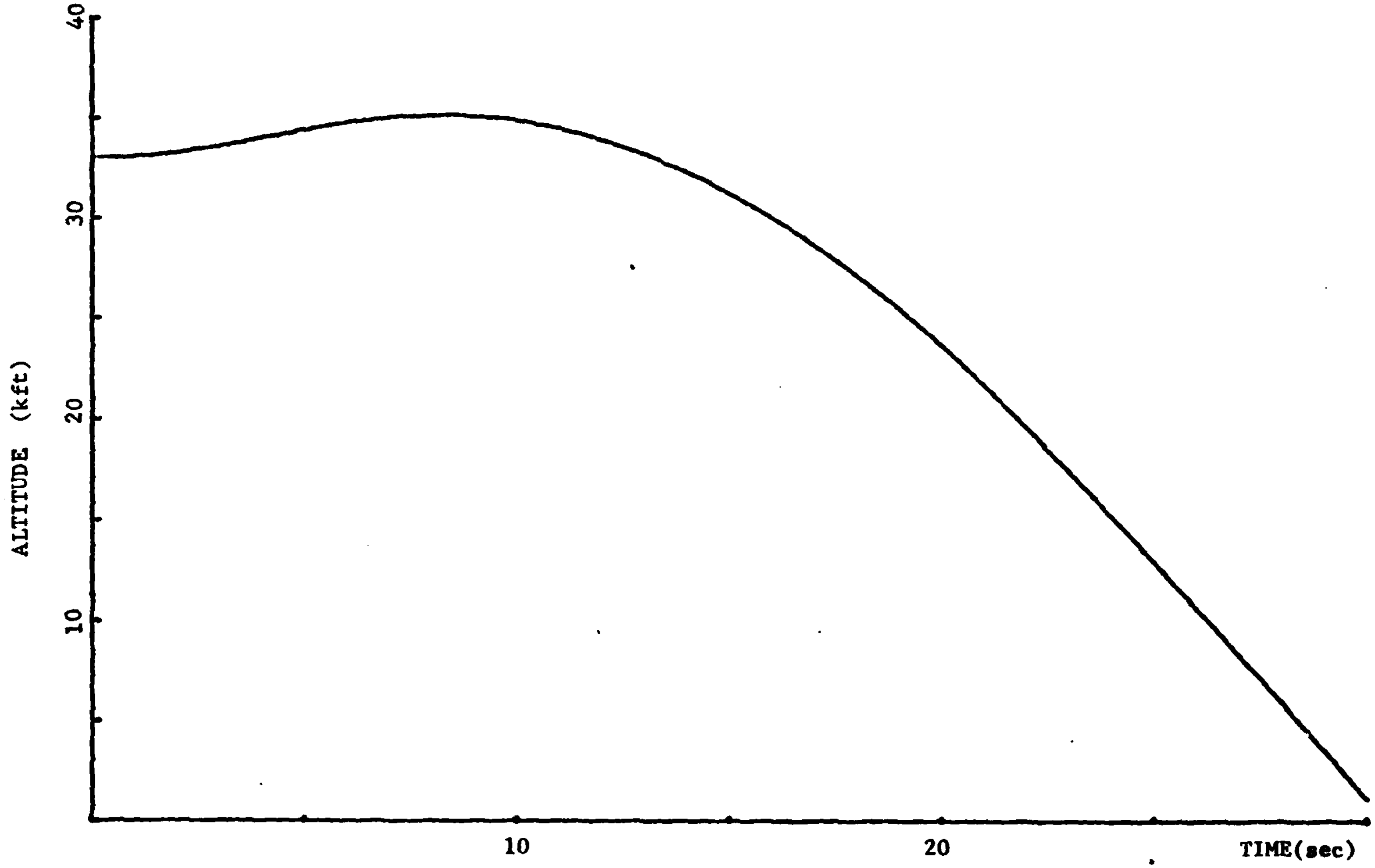
TABLE VIII

inputed as 2000.0 feet. As the simulation progressed, the missile developed the necessary commands to climb and regain the desired cruise altitude (Figure 15). It then commenced its terminal dive. The target was detected, acquired, and impact followed. This initial condition modification produced the largest miss distance (0.457 ft.).

Figure 16 was presented to show the relative locations of the miss distances for each change. Table VIII reveals that up until 1800 feet, changes to YTP created very little variation in the miss distance. It should be noted that the larger the displacement became, the greater the time until impact. The increase in time was necessary to allow the missile to acquire the target and then compute the required acuator commands to impact the target. This concluded the investigation of the effects of initial displacement error on target miss distance.

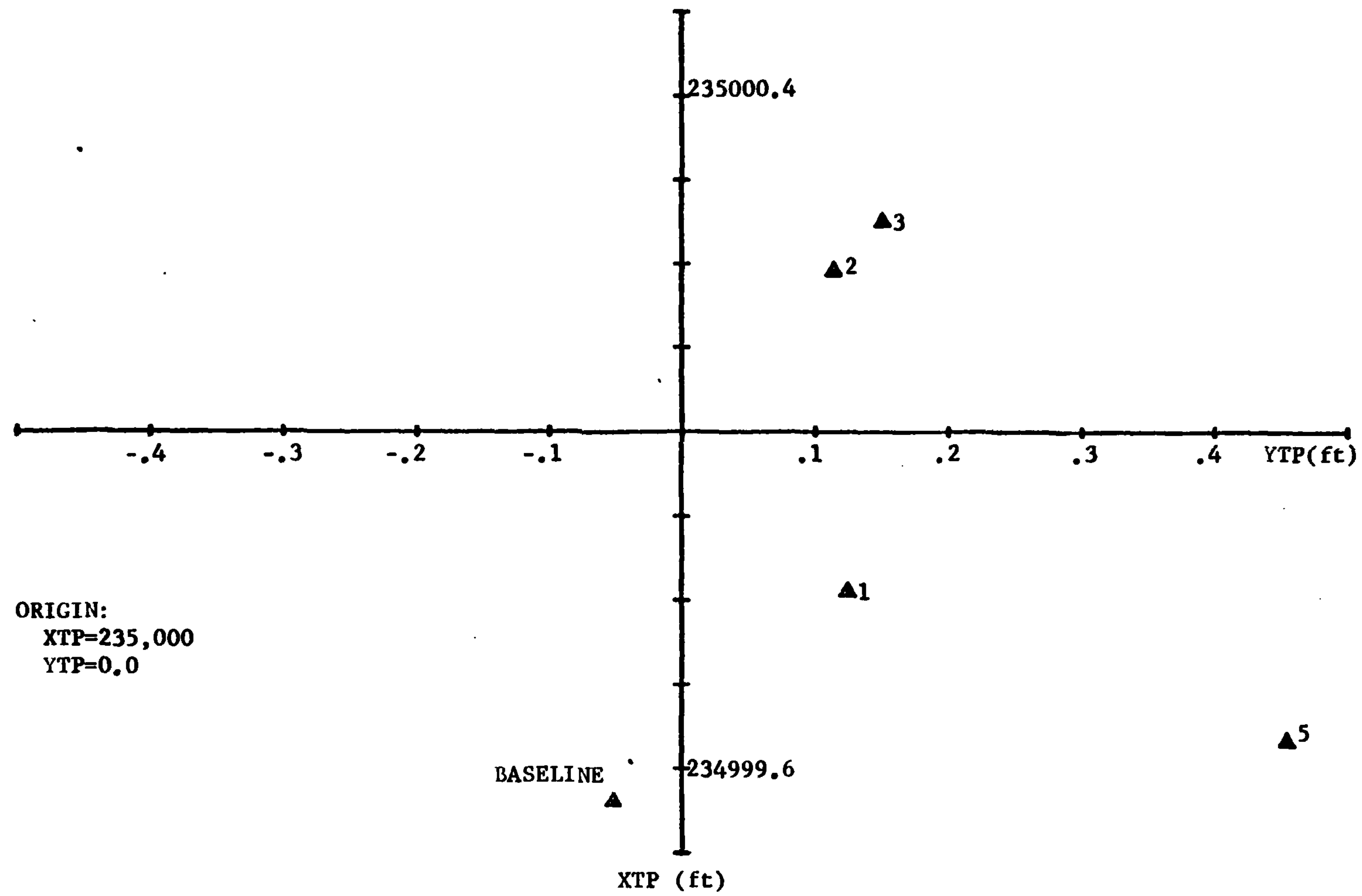
The idea behind changing the cruise altitude was to try and simulate a sea skimming profile. RAMJET contains tables that allow for four cruise altitudes. Of these four altitudes, the lowest (500 ft) was chosen even though it is high for a sea-skimmer. To run this simulation only two input data card changes were required. HO(414) and HREF(501) had to be set equal to the desired altitude. The resultant flight path is shown in Figure 17.

The simulation produced an error-free output, but at first glance the results appeared unacceptable. Still trying to compare miss distances, the downrange distance (XTP) was found to be 197916 feet. Comparing this to the terminal baseline (234999) resulted in an extreme error. It was then realized that the missile must expend more fuel at this altitude to attain the same speed. Therefore, the results could be correct. As further verification of the correctness of the run, it was assumed that the missile weight at impact should be fairly equal. The



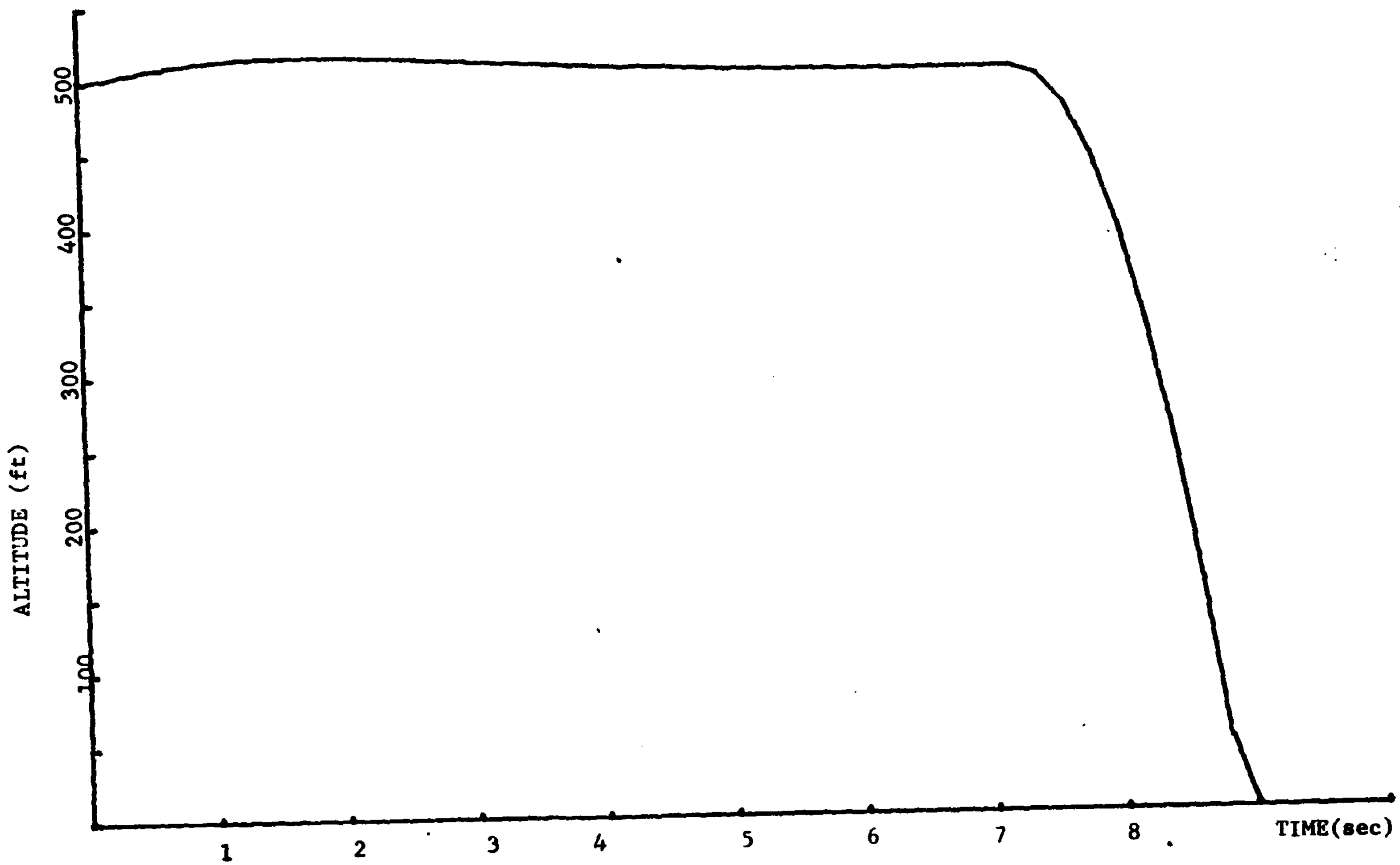
ALTITUDE VS. TIME (TERMINAL - ZTP)

FIGURE 15



IMPACT RESULTS

FIGURE 16



ALTITUDE VS. TIME (SEA-SKIMMER)

FIGURE 17

simulation impact weight was 1126.34 lbs while the terminal baseline weighed 1113.12 lbs. Since no conflicting information could be found in reference 12, it was concluded that this simulation was correct.

The final flight disturbance investigated was the addition of random noise to the missile homing seeker. This simulation proved to be unsuccessful due to invalid input format. The information describing the process is brief and difficult to understand. Further information from NWC China Lake is required to be able to successfully run simulations with random noise.

With the completion of these modifications, many unanswered problems and questions still exist. The two most critical problems concern the input of random noise generators and the missile exceeding the angle of attack limitation when using ENGINE for cruise propulsion. However, the simulations did show that the MOD6DF program runs correctly at alternate cruise altitudes and with initial displacement errors.

VI. CONCLUSION

The MOD6DF computer program from NWC China Lake was converted to operate on the IBM-360 computer at the Naval Postgraduate School. The program would only function properly when using the simplified ramjet model. When this model was not used, the missile angle of attack exceeded the maximum limit of ten degrees. This error caused the ramjet engine to flame out.

Target impact errors for the terminal guidance problem were investigated when the initial displacement was modified. These modiifications demonstrate the missile's accuracy when removed from the ideal flight path. The results also point out that if the target falls within the seeker search pattern, target impact will inevitably happen.

Random noise generation is possible with the MOD6DF program. However, more information discussing the parameters required to develop the noise is necessary. Additionally, sample noise inputs should be obtained that reflect the alteration to the noise subroutines.

This research involved the preliminary investigation of the MOD6DF program. The program has many possible areas, concerning guidance and control of tactical missiles, which could be developed for future study. These areas not only include the unresolved problems encountered during this research. Additionally, reference 14 and 15 contain many examples of possible advanced guidance concepts.

COMPUTER OUTPUT

1	INPT1	[illegible]	0.0	0.0
1	CUPT2.3	[illegible]	0.0	0.0
1	STGE2.3	[illegible]	0.0	0.0
2	MODULE G1I,G1	[illegible]	0.0	0.0
2	MODULE G3I,G3	[illegible]	0.0	0.0
2	MODULE G5I,G5	[illegible]	0.0	0.0
2	MODULE S5I,S5	[illegible]	0.0	0.0
2	MODULE S2I,S2	[illegible]	0.0	0.0
2	MODULE S1I,S1	[illegible]	0.0	0.0
2	MODULE C2I,C2	[illegible]	0.0	0.0
2	MODULE C1I,C1	[illegible]	0.0	0.0
2	MODULE A4I,A4	[illegible]	0.0	0.0
2	MODULE A3I,A3	[illegible]	0.0	0.0
2	MODULE A2I,A2	[illegible]	0.0	0.0
2	MODULE A1I,A1	[illegible]	0.0	0.0
2	MODULE [illegible]	15	0.0	0.0
2	MODULE [illegible]	17	0.0	0.0
3	W	[illegible]	[illegible] 01	[illegible] 01
3	FLCSTF	[illegible]	[illegible] 01	0.0
3	S	[illegible]	[illegible]	0.0
3	CHAR	[illegible]	[illegible] 01	0.0
3	CGI	[illegible]	0.9464465[illegible] 01	0.1000000[illegible] 01
3	EPSTH	[illegible]	0.0	0.0
3	NUTH	[illegible]	0.0	0.0
3	A	201	0.6214459[illegible] 01	0.1000000[illegible] 01
3	B	[illegible]	0.4[illegible] 01	0.1000000[illegible] 01
3	CC	[illegible]	0.4[illegible] 03	0.10000000E 01
3	TSA	[illegible]	0.0	0.0
3	ELP	210	0.99999993E-03	0.0
3	ELP	211	0.99999964E-01	0.0
3	P	212	0.0	0.10000000E 01
3	ELQ	214	0.99999993E-03	0.0
3	ELQ	215	0.99999964E-01	0.0
3	Q	217	0.0	0.10000000E 01
3	ELR	218	0.99999993E-03	0.0
3	ELR	219	0.99999964E-01	0.0
3	R	[illegible]	0.0	0.10000000E 01
3	ELA11	[illegible]	0.99999993E-03	0.0
3	ELA11	[illegible]	0.99999964E-01	0.0
3	ELA12	[illegible]	0.99999993E-03	0.0
3	ELA12	[illegible]	0.99999964E-01	0.0
3	ELA13	[illegible]	0.99999993E-03	0.0
3	ELA13	[illegible]	0.99999964E-01	0.0
3	ELA21	[illegible]	0.99999993E-03	0.0
3	ELA21	[illegible]	0.99999964E-01	0.0
3	ELA22	[illegible]	0.99999993E-03	0.0
3	ELA22	[illegible]	0.99999964E-01	0.0
3	ELA23	[illegible]	0.99999993E-03	0.0
3	ELA23	[illegible]	0.99999964E-01	0.0
3	ELA31	[illegible]	0.99999993E-03	0.0
3	ELA31	[illegible]	0.99999964E-01	0.0
3	ELA32	[illegible]	0.99999993E-03	0.0
3	ELA32	[illegible]	0.99999964E-01	0.0
3	ELA33	[illegible]	0.99999993E-03	0.0
3	ELA33	[illegible]	0.99999964E-01	0.0
3	CG139	[illegible]	0.9[illegible] 01	0.0
3	AKS	260	0.20[illegible] 02	0.0
3	OMGYO	[illegible]	0.0	0.10000000E 01
3	ELOMGY	[illegible]	0.99999993E-03	0.0
3	ELOMGY	264	0.99999964E-01	0.0
3	OMGY	265	0.0	0.10000000E 01
3	OMGZO	266	0.0	0.10000000E 01
3	ELOMGZ	267	0.99999993E-03	0.0
3	ELOMGZ	268	0.99999964E-01	0.0
3	OMGZ	269	0.0	0.10000000E 01
3	THTSO	270	0.0	0.10000000E 01
3	ELTHTS	271	0.99999993E-03	0.0
3	ELTHTS	272	0.99999964E-01	0.0
3	THTS	273	0.0	0.10000000E 01
3	PSISO	274	0.0	0.10000000E 01
3	ELPSIS	275	0.99999993E-03	0.0
3	ELPSIS	276	0.99999964E-01	0.0
3	PSIS	277	0.0	0.10000000E 01
3	AG	[illegible]	0.2171[illegible]97[illegible] 02	0.0
3	ELVX	284	0.99999993E-03	0.0
3	ELVX	285	0.99999964E-01	0.0
3	VX	286	0.4[illegible] 04	0.10000000E 01
3	ELX S	[illegible]	0.99999993E-03	0.0
3	ELX S	289	0.99999964E-01	0.0
3	X	290	0.1[illegible] 06	0.10000000E 01
3	ELVY S	292	0.99999993E-03	0.0
3	ELVY S	293	0.99999964E-01	0.0
3	VY	294	0.0	0.10000000E 01

```
3   [illegible]3          1715   [illegible]-01  0.0
3   D3                    [illegible]   0.0             0.10000000E 01
3   [illegible]4          [illegible]   [illegible]-03  0.0
3   [illegible]4          [illegible]   [illegible]-01  0.0
3   D4                    [illegible]   0.0             0.10000000E 01
3   [illegible]Z1         [illegible]   [illegible]-03  0.0
3   [illegible]Z1         [illegible]   [illegible]-01  0.0
3   Z1                    [illegible]   0.0             0.10000000E 01
3   [illegible]LY1        [illegible]   [illegible]-03  0.0
3   [illegible]LY1        [illegible]   [illegible]-01  0.0
3   Y1                    [illegible]   0.0             0.10000000E 01
3   [illegible]GAS        [illegible]   [illegible]     0.0
3   [illegible]P S        [illegible]   [illegible] 00  0.0
3   [illegible] FIN       [illegible]   [illegible]     0.0
3   [illegible] FIN       [illegible]   [illegible]     0.0
3   MAX FIN               [illegible]   [illegible] 00  0.0
3   [illegible]           [illegible]   [illegible] 01  0.0
3   [illegible]           [illegible]   [illegible] 00  0.0
3   [illegible]           [illegible]   0.0             0.0
3   PITCH[illegible]      [illegible]   [illegible]-01  0.0
3   [illegible]           [illegible]   0.0             0.0
3   [illegible]           [illegible]   [illegible] 02  0.0
3   [illegible]           [illegible]   [illegible] 01  0.0
3   [illegible]           [illegible]   [illegible]-02  0.0
3   [illegible]           [illegible]   [illegible]-01  0.0
3   [illegible]           [illegible]   [illegible]-02  0.10000000E 01
3   [illegible]           34[illegible] 0.0             0.0
4       THRUST            [illegible]   0.0             0.0
4       [illegible]       [illegible]   0.0             0.0
4       X(TP)             [illegible]   0.0             0.0
4       VX(TP)            [illegible]   0.0             0.0
4       [illegible]       [illegible]   0.0             0.0
4       [illegible]       [illegible]   0.0             0.0
4       ALT               [illegible]   0.0             0.0
4       Y(TP)             [illegible]   0.0             0.0
4       VY(TP)            [illegible]   0.0             0.0
4       [illegible]       16[illegible] 0.0             0.0
4       [illegible]       [illegible]   0.0             0.0
4       VELOCITY          [illegible]   0.0             0.0
4       Z(TP)             [illegible]   0.0             0.0
4       VZ(TP)            [illegible]   0.0             0.0
4       [illegible]       [illegible]   0.0             0.0
4       [illegible]       [illegible]   0.0             0.0
4       PITCH             [illegible]   0.0             0.0
4       ALPHA(DEG)        [illegible]   0.0             0.0
4       [illegible]       16[illegible] 0.0             0.0
4       [illegible]       [illegible]   0.0             0.0
4       [illegible]       16[illegible] 0.0             0.0
4       [illegible]       [illegible]   0.0             0.0
4       DELTA [illegible] [illegible]   0.0             0.0
4       DELTA [illegible] [illegible]   0.0             0.0
4       DELTA [illegible] [illegible]   0.0             0.0
4       [illegible] MARGIN [illegible]  0.0             0.0
4       [illegible]       [illegible]   0.0             0.0
4       YAW               [illegible]   0.0             0.0
4       [illegible]       [illegible]   0.0             0.0
4       GAMMA[illegible]  [illegible]   0.0             0.0
6                         [illegible]   0.0             0.0
E[illegible]NT
[illegible] 0.4[illegible]7 , [illegible]9 , [illegible]9 , 1.52749997 , 0.6521[illegible] , 0.1147[illegible] , [illegible] , 0.9[illegible]97 ,
 1.[illegible] , 2.[illegible] , [illegible]0 , 1.1999998 , 1.36[illegible]-01, 14.63[illegible] , [illegible] , 0.0 ,
 0.0 , 0.0 , 0.[illegible] , 0.0 , 0.0 , 0.0 , [illegible] , [illegible] ,
 0.0 , 0.0 , 1.[illegible] , 0.1 , 0.0 , 0.0 , [illegible]
 2. [illegible] 2. 0. 0. 1. 0. [illegible] 0. 0. 1. 0.
 0. [illegible]
END
    *** [illegible] STANDARD ATMOSPHERE [illegible]
    *** ALTITUDE [illegible] 30480. METERS
```

TIME	THRUST C F P ANYC INLET MARGIN	WEIGHT ALT VELOCITY PITCH AMACH DYN P	X(TP) Y(TP) Z(TP) ALPHA(DEG) DELTA E(DEG) YAW	[illegible] [illegible] [illegible] [illegible] DELTA [illegible] ROLL	[illegible] [illegible] [illegible] BETA(DEG) DELTA F(DEG) [illegible]
0.49999950E-02	0.11416750E 04 0.25043402E-01 0.0 0.0 0.0 0.0	0.11525[illegible] 04 0.35000000E 05 0.2750[illegible]E 04 0.30764110E-01 0.28195744E 01 0.19290527E 02	0.17356625E 06 0.0 0.45068915E-04 0.17630310E 01 0.86758852E-01 0.0	[illegible] 04 0.0 [illegible]-01 [illegible] 00 0.0 0.0	0.0 -0.49994611E [illegible] -0.2866454[illegible] 0.0 0.0 -0.66045755E-[illegible]
0.[illegible]999952E 00	0.11322258E 04 -0.52350126E-02 -0.80484524E-03 -0.77175372E-02 -0.91046725E-14 0.0	0.11523550E 04 0.35011273E 05 0.27502771E 04 0.42267656E-01 0.2815987[illegible] 01 0.19290421E 02	0.17413070E 06 0.51678164E-03 -0.27663856E 01 0.23562231E 01 0.65386245 01 -0.58246452 -04	[illegible] 04 [illegible]-01 [illegible] 01 [illegible] 00 [illegible]-02 [illegible]-03	0.10[illegible] -0.5162126[illegible] -0.5[illegible] [illegible] 0.1[illegible]-01 0.1243622[illegible]
0.40499951E 00	0.11401408E 04 -0.5[illegible]E-01 0.3270747[illegible]-03 0.78449851E-02 -0.18125553E-03 0.0	0.11521667E 04 0.35001301E 05 0.2750035[illegible] 04 0.3505145[illegible]-01 0.2819662[illegible] 01 0.19289764E 02	0.17466625E 06 0.27381351E-02 -0.13016148E 01 0.1871[illegible]538E 01 0.51719565E 01 -0.13072976E-03	[illegible] 04 [illegible]-02 [illegible] 01 [illegible] [illegible]-01 [illegible]-03	-0.153[illegible] -0.62[illegible] -0.4[illegible] 0.6[illegible] -0.7[illegible] 0.2[illegible]
0.60499984E 00	0.11444[illegible] 04 -0.35235365E-01 0.42343186E-03 -0.6256113[illegible]-04 0.17055035E-03 0.0	0.11519714E 04 0.35002770E 05 0.27504165E 04 0.36817691E-01 0.2826665[illegible] 01 0.1929625[illegible] 02	0.17521685E 06 -0.22553256E-02 -0.27724276E 01 0.1319628[illegible] 01 0.43705068E 01 -0.35515150E-04	[illegible] 04 [illegible]-01 [illegible] [illegible] 00 [illegible]-02 [illegible]-04	-0.3[illegible]-01 -0.3[illegible] -0.31[illegible] 0.1[illegible] -0.7[illegible] 0.[illegible]
0.80499983E 00	0.11462571E 04 -0.20372777E-01 0.10[illegible]-03 -0.10566664E-03 0.44583622E-03 0.0	0.11517761E 04 0.35004240E 05 0.27506595E 04 0.19836720E-01 0.28205580E 01 0.19300659E 02	0.17576644E 06 -0.46110637E-02 -0.47475067E 01 0.99310791E 00 0.3955[illegible] 01 0.24477573E-04	[illegible] 04 [illegible]-01 [illegible] [illegible] [illegible]-02 [illegible]-04	0.4[illegible]-01 -0.2[illegible] -0.[illegible] -0.1[illegible] 0.1[illegible] 0.[illegible]

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

COMPUTER PROGRAM

```
C
C
C        THIS IS THE MAIN MCD6CF PROGRAM
C
      COMMON C(3415), TEMPS(1500)                                       MCD60010
      EQUIVALENCE (C(3315),N)                                           MCD60020
      EQUIVALENCE (C(2911),HMIN)                                        MCD60030
      EQUIVALENCE (C(2912),HMAX)                                        MCD60040
      EQUIVALENCE (C(2913),DER(1))                                      MCD60050
      EQUIVALENCE (C(3014),VAR(1))                                      MCD60060
      EQUIVALENCE (C(3115),EL(1))                                       MCD60070
      EQUIVALENCE (C(3215),EU(1))                                       MCD60100
      EQUIVALENCE (C(3316),IPL(1))                                      MCD60110
      EQUIVALENCE (C( 9321),T       )                                   MCD60120
      EQUIVALENCE (C(2904),KSTEP), (C(2905),STEP)                       MCD60130
      EQUIVALENCE(C(2907),LSTEP)                                        MCD60140
      DIMENSION IPL(100)                                                MCD60150
      DIMENSION DER(101)                                                MCD60160
      DIMENSION VAR(101)                                                MCD60170
      DIMENSION  EL(100)                                                MCD60200
      DIMENSION  EU(100)                                                MCD60210
      M=N
      N=M
 1000 CALL ZERO                                                         MCD60220
 1001 CALL CINPT1                                                       MCD60230
      LSTEP = STEP                                                      MCD60240
 1002 CALL  SLEL1                                                       MCD60250
 1003 CALL  AUXI                                                        MCD60260
 1004 CALL  SLEL2                                                       MCD60270
 1005 DO 60 I = 2,N                                                     MCD60300
      J = IPL(I-1)                                                      MCD60310
      EL(I-1) = C(J+1)                                                  MCD60320
      EU(I-1) = C(J+2)                                                  MCD60330
      VAR(I) = C(J+3)                                                   MCD60340
   60 DER(I) = C(J)                                                     MCD60350
      VAR(1) = T                                                        MCD60360
 1006 CALL   AUXSUB                                                     MCD60370
 1007 CALL AMFK                                                         MCD60400
 1008 DO 50 I = 2, N                                                    MCD60410
      J = IPL(I-1)                                                      MCD60420
   50 C(J+3) = VAR(I)                                                   MCD60430
      T = VAR(1)                                                        MCD60440
 1009 CALL  SLEL3                                                       MCD60450
C     IF (KSTEP .EQ. 1) GO TO 1007                                      MCD60460
      IF (KSTEP - 1) 70,1007,70                                         MCD60470
   70 CALL PROCES                                                       MCD60500
      CALL RESET                                                        MCD60510
      GO TO (1000,1001,1002,1003,1004,1005,1006,1007,1008,1009,1010),   MCD60520
```

```
    1 LSTEP                                                              MCD60530
 1010 CALL EXIT                                                          MCD60540
      STOP                                                               MCD60545
      END                                                                MCD60550
C
C
C
      SUBROUTINE ZERO
C
      COMMON C(3415)
      DO 1 KLEAR = 1, 4915
    1 C(KLEAR) = 0.0
      RETURN
      END
C
C
      SUBROUTINE OINPT1                                                  CINP0030
C
C     BASIC INPUT SUBROUTINE CINPT1                                      CINP0010
C     SUBROUTINE OINPT1 IS THE BASIC INPUT SUBROUTINE                    CINP0020
C
      DIMENSION LISTNO(50), VALUE(50)                                    CINP0230
      DIMENSION SUBNO(99),IR(2),VR(2)                                    CINP0240
      DIMENSION RNDMNO(50)                                               CINP0250
      DIMENSION ALPHA(4),CNAME1(50),ONAME2(50),ONAME3(50),OUTNO(50)
      DIMENSION MODNO(99)
      DIMENSION STATNO (100)                                             CINP0300
      COMMON C(3415)                                                     CINP0040
      REAL MCONC                                                         CINP0310
      INTEGER OUTNO                                                      CINP0320
      INTEGER RNDMNO                                                     CINP0330
      INTEGER STATNO                                                     CINP0340
      EQUIVALENCE (C(2442),LOSTAT)                                       CINP0050
      EQUIVALENCE (C(2200),STATNO(1)),(C(2441),NOSTAT)
      EQUIVALENCE  (C(2801),NOSUB),(C(2802),SUBNO(1)),(C(2663),IR(1)),
     1 (C(2665),VR(1))
      EQUIVALENCE   (C(2701),NCMOD)                                      CINP0110
      EQUIVALENCE   (C(2661),NCOUT)                                      CINP0120
      EQUIVALENCE ( C(2448),   NORNDM )                                  CINP0130
      EQUIVALENCE  (C(2300),NOLIST),(C(2301),LISTNO(1)),(C(2351),
     1 VALUE(1))
      EQUIVALENCE  (C(2500) , OUTNO(1))
      EQUIVALENCE  (C(2550) , CNAME1(1))
      EQUIVALENCE  (C(2600) , ONAME2(1))
      EQUIVALENCE (C(2150) , ONAME3(1))
      EQUIVALENCE  (C(2450) , RNDMNO(1))
      EQUIVALENCE (C(2702),MODNO(1))
      JAR = 0                                                            CINP0350
```

```
      1 READ(5,2) IR(1),ALPHA(1),ALPHA(2),ALPHA(3),ALPHA(4),IR(2),VR(1),
     1 VR(2)
        WRITE(6,2) IR(1),ALPHA(1),ALPHA(2),ALPHA(3),ALPHA(4),IR(2),VR(1),
     1 VR(2)
      2 FORMAT(I2,2X,4A4,I5,5X,2E15.8)
C       IF (IR(1) .NE. 0) GO TO 7                                       CINP0410
        IF (IR(1)) 7,18,7                                               CINP0420
     18 I1 = IR(2)                                                      CINP0430
        REWIND I1                                                       CINP0440
        I2 = VR(1) - 1.                                                 CINP0450
        I3 = VR(2)                                                      CINP0460
        IF (I2) 11,15,14                                                CINP0470
     14 DO 8 I = 1,I2                                                   CINP0500
      8 READ (I1,10) J,X                                                CINP0510
     15 DO 9 I = 1,I3                                                   CINP0520
        READ (I1,2) J,X                                                 CINP0530
      9 C(J) = X                                                        CINP0540
     10 FORMAT (I5,E15.9)                                               CINP0550
        GO TO 1                                                         CINP0560
     11 JAR = 1                                                         CINP0570
        GO TO 1                                                         CINP0600
C     7 IF (IR(1) .NE. 1) GO TO 3                                       CINP0610
      7 IF (IR(1) - 1) 3,19,3                                           CINP0620
     19 NOSUB = NOSUB + 1                                               CINP0630
        SUBNO(NOSUB) = IR(2)                                            CINP0640
        GO TO 1                                                         CINP0650
C     3 IF (IR(1) .NE. 2) GO TO 4                                       CINP0660
      3 IF (IR(1) - 2) 4,20,4                                           CINP0670
     20 NOMOD = NOMOD + 1                                               CINP0700
        MODNO(NOMOD) = IR(2)                                            CINP0710
        GO TO 1                                                         CINP0720
C     4 IF (IR(1) .NE. 3) GO TO 5                                       CINP0730
      4 IF (IR(1) - 3) 5,21,5                                           CINP0740
     21 L = IR(2)                                                       CINP0750
        C(L) = VR(1)                                                    CINP0760
C       IF (JAR .EQ. 1) WRITE (I1,10)L,VR(1)                            CINP0770
        IF (JAR - 1) 23,22,23                                           CINP1000
     22 WRITE (I1,10)L,VR(1)                                            CINP1010
C       IF (VR(2) .EQ. 0.) GO TO 1                                      CINP1020
     23 IF (VR(2)) 24,1,24                                              CINP1030
     24 NOLIST = NOLIST + 1                                             CINP1040
        LISTNO(NOLIST) = L                                              CINP1050
        VALUE(NOLIST) = VR(1)                                           CINP1060
        GO TO 1                                                         CINP1070
C     5 IF (IR(1) .NE. 4) GO TO 6                                       CINP1100
      5 IF (IR(1) - 4) 6,25,6                                           CINP1110
     25 NOCUT = NOCUT +1                                                CINP1120
        CNAME1(NOCUT) = ALPHA(2)
```

```
      CNAME2(NCCUT) = ALPHA(3)
      CNAME3(NOCUT) = ALPHA(4)
      CUTNC(NCCUT) = IR(2)                                              CINP1150
      GO TC 1                                                           CINP116C
C   6 IF (IR(1) .NE. 5) GC TO 16                                        CINP1170
    6 IF (IR(1) - 5) 16,26,16                                           CINP1200
C     IF (VR(1) .EQ. 0.) GC TC 17                                       CINP1210
   26 IF (VR(1)) 27,17,27                                               CINP1220
   27 LOSTAT = LCSTAT + 1                                               CINP123C
   17 NOSTAT = NOSTAT + 1                                               CINP1240
      STATNO(NOSTAT) = IR(2)                                            CINP1250
      GO TC 1                                                           CINP126C
   16 NORNDM=C
C     IF (IR(2).EQ.0) RETURN
      IF (IR(2)) 29,28,29
   28 RETURN                                                            CINP1310
   29 N = IR(2)                                                         CINP132C
      DO 12 I = 1, N                                                    CINP133C
      READ (5,13) J,Y,ZSTART,SIGNO,BETA
      WRITE (6,13) J,Y,ZSTART,SIGNO,BETA
   13 FORMAT (I5,F5.0,3E15.9)
      NORNDM = NORNDM + 1                                               CINP1370
      RNDMNC(I) = J                                                     CINP140C
      C(J) = Y                                                          CINP141C
      C(J+1)=ZSTART
      C(J+3) = SIGNO                                                    CINP144C
   12 C(J+4) = BETA                                                     CINP1450
      RETURN                                                            CINP146C
      END                                                               CINP147C
C
C
C
      SUBROUTINE   SUBL1                                                SUBL0010
C
      COMMON C(3415)                                                    SUBL002C
      DIMENSION  SUBNO(99)                                              SUBL0050
      EQUIVALENCE (C(2801),NOSUB)                                       SUBL003C
      EQUIVALENCE  (C(2802),SUBNO(1))
      DO 1 I = 1, NOSUB                                                 SUBL0060
      J = SUBNO(I)                                                      SUBL0070
      GO TO ( 1, 2, 3, 4, 5, 6, 7, 8, 9 ), J                            SUBL0100
    2 CALL INFT1                                                        SUBL011C
      GO TC 1                                                           SUBL012C
    3 CALL CLFT1                                                        SUBL0130
      GO TC 1                                                           SUBL014C
    4 CALL STGE1                                                        SUBL0150
      GO TC 1                                                           SUBL0160
    5 CALL CNTF1                                                        SUBL017C
```

```
      GO TO 1                                                           SUBL0200
    6 CALL RNCM1                                                        SUBL0210
      GO TO 1                                                           SUBL0220
    7 CALL ALXA1                                                        SUBL0230
      GO TO 1                                                           SUBL0240
    8 CALL ALXB1                                                        SUBL0250
      GO TO 1                                                           SUBL0260
    9 CALL ALXC1                                                        SUBL0270
    1 CONTINUE                                                          SUBL0300
      RETURN                                                            SUBL0310
       END                                                              SUBL0320
C
C
C
      SUBROUTINE   SUBL2                                                SUBL0010
C
      DIMENSION  SUBNO(99)                                              SUBL0050
      COMMON C(3415)                                                    SUBL0020
      EQUIVALENCE (C(2801),NCSUB)                                       SUBL0030
      EQUIVALENCE  (C(2802),SUBNO(1))
      DO 1 I = 1, NCSUB                                                 SUBL0060
      J = SUBNO(I)                                                      SUBL0070
      GO TO ( 1, 2, 3, 4, 5, 6, 7, 8, 9 ), J                            SUBL0100
    2 CALL INFT2                                                        SUBL0110
      GO TO 1                                                           SUBL0120
    3 CALL CLFT2                                                        SUBL0130
      GO TO 1                                                           SUBL0140
    4 CALL STGE2                                                        SUBL0150
      GO TO 1                                                           SUBL0160
    5 CALL CNTR2                                                        SUBL0170
      GO TO 1                                                           SUBL0200
    6 CALL RNCM2                                                        SUBL0210
      GO TO 1                                                           SUBL0220
    7 CALL ALXA2                                                        SUBL0230
      GO TO 1                                                           SUBL0240
    8 CALL ALXB2                                                        SUBL0250
      GO TO 1                                                           SUBL0260
    9 CALL ALXC2                                                        SUBL0270
    1 CONTINUE                                                          SUBL0300
      RETURN                                                            SUBL0310
       END                                                              SUBL0320
C
C
C
      SUBROUTINE   SUBL3                                                SUBL0010
C
      DIMENSION  SUBNO(99)                                              SUBL0050
      COMMON C(3415)                                                    SUBL0020
```

```
      EQUIVALENCE (C(2801),NOSUB)                                       SLBL0030
      EQUIVALENCE  (C(2802),SUBNO(1))                                   SLBL0060
      DO 1 I = 1, NOSUB                                                 SLBL0070
      J = SUBNO(I)                                                      SLBL0100
      GO TO ( 1, 2, 3, 4, 5, 6, 7, 8, 9 ), J                            SLBL0110
    2 CALL INPT3                                                        SLBL0120
      GO TO 1                                                           SLBL0130
    3 CALL CLFT3                                                        SLBL0140
      GO TO 1                                                           SLBL0150
    4 CALL STGE3                                                        SLBL0160
      GO TO 1                                                           SLBL0170
    5 CALL CNTR3                                                        SLBL0200
      GO TO 1                                                           SLBL0210
    6 CALL RNDM3                                                        SLBL0220
      GO TO 1                                                           SLBL0230
    7 CALL ALXA3                                                        SLBL0240
      GO TO 1
    8 CALL ALXB3                                                        SLBL0250
      GO TO 1                                                           SLBL0260
    9 CALL ALXC3                                                        SLBL0270
    1 CONTINUE                                                          SLBL0300
      RETURN                                                            SLBL0310
       END                                                              SLBL0320
C
C
C
      SUBROUTINE ALXI                                                   ALXI0010
C
      DIMENSION  MCDNO(99)                                              ALXI0060
      COMMON C(3415)                                                    ALXI0020
      REAL MCDNO                                                        ALXI0070
      EQUIVALENCE (C(2701), NCMOD)                                      ALXI0030
      EQUIVALENCE (C(2702),MCDNO(1))
      EQUIVALENCE (C(2659), CNCE)
      EQUIVALENCE (C(3315),N)                                           ALXI0050
      N = 1                                                             ALXI0100
      CNCE = 0
      DO 1 I=1,NCMOD                                                    ALXI0110
      L = MCDNO(I)                                                      ALXI0120
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23ALXI0130
     1      ,24,25,26,27,28,29,30,31,32,33,34,35,36,37),L               ALXI0140
    2 CALL A1I                                                          ALXI0150
      GO TO 1                                                           ALXI0160
    3 CALL A2I                                                          ALXI0170
      GO TO 1                                                           ALXI0200
    4 CALL A3I                                                          ALXI0210
      GO TO 1                                                           ALXI0220
```

```
    5 CALL A4I                                                          ALXI0230
      GO TO 1                                                           ALXI0240
    6 CALL A5I                                                          ALXI0250
      GO TO 1                                                           ALXI0260
    7 CALL C1I                                                          ALXI0270
      GO TO 1                                                           ALXI0300
    8 CALL C2I                                                          ALXI0310
      GO TO 1                                                           ALXI0320
    9 CALL C3I                                                          ALXI0330
      GO TO 1                                                           ALXI0340
   10 CALL C4I                                                          ALXI0350
      GO TO 1                                                           ALXI0360
   11 CALL C5I                                                          ALXI0370
      GO TO 1                                                           ALXI0400
   12 CALL C6I                                                          ALXI0410
      GO TO 1                                                           ALXI0420
   13 CALL C7I                                                          ALXI0430
      GO TO 1                                                           ALXI0440
   14 CALL C8I                                                          ALXI0450
      GO TO 1                                                           ALXI0460
   15 CALL C9I                                                          ALXI0470
      GO TO 1                                                           ALXI0500
   16 CALL C10I                                                         ALXI0510
      GO TO 1                                                           ALXI0520
   17 CALL D1I                                                          ALXI0530
      GO TO 1                                                           ALXI0540
   18 CALL D2I                                                          ALXI0550
      GO TO 1                                                           ALXI0560
   19 CALL D3I                                                          ALXI0570
      GO TO 1                                                           ALXI0600
   20 CALL D4I                                                          ALXI0610
      GO TO 1                                                           ALXI0620
   21 CALL D5I                                                          ALXI0630
      GO TO 1                                                           ALXI0640
   22 CALL G1I                                                          ALXI0650
      GO TO 1                                                           ALXI0660
   23 CALL G2I                                                          ALXI0670
      GO TO 1                                                           ALXI0700
   24 CALL G3I                                                          ALXI0710
      GO TO 1                                                           ALXI0720
   25 CALL G4I                                                          ALXI0730
      GO TO 1                                                           ALXI0740
   26 CALL G5I                                                          ALXI0750
      GO TO 1                                                           ALXI0760
   27 CALL G6I                                                          ALXI0770
      GO TO 1                                                           ALXI1000
   28 CALL S1I                                                          ALXI1010
      GO TO 1                                                           ALXI1020
```

```
   29 CALL S2I                                                          ALXI1030
      GO TO 1                                                           ALXI1040
   30 CALL S3I                                                          ALXI1050
      GO TO 1                                                           ALXI1060
   31 CALL S4I                                                          ALXI1070
      GO TO 1                                                           ALXI1100
   32 CALL S5I                                                          ALXI1110
      GO TO 1                                                           ALXI1120
   33 CALL S6I                                                          ALXI1130
      GO TO 1                                                           ALXI1140
   34 CALL S7I                                                          ALXI1150
      GO TO 1                                                           ALXI1160
   35 CALL S8I                                                          ALXI1170
      GO TO 1                                                           ALXI1200
   36 CALL S9I                                                          ALXI1210
      GO TO 1                                                           ALXI1220
   37 CALL S10I                                                         ALXI1230
    1 CONTINUE                                                          ALXI1240
      RETURN                                                            ALXI1250
      END                                                               ALXI1260
C
C
C
      SUBROUTINE AUXSUB                                                 ALXS0010
C
      DIMENSION DER(101)                                                ALXS0120
      DIMENSION VAR(101)                                                ALXS0130
      DIMENSION IPL(100)                                                ALXS0140
      DIMENSION  MODNO(99)                                              ALXS0150
      COMMON C(3415)                                                    ALXS0020
      REAL MODNO                                                        ALXS0160
      EQUIVALENCE (C(2701), NOMOD)                                      ALXS0030
      EQUIVALENCE (C(2702),MODNO(1))
      EQUIVALENCE (C(3316),IPL(1))
      EQUIVALENCE (C(3014),VAR(1))
      EQUIVALENCE (C(2913),DER(1))
      EQUIVALENCE (C( 932),T)                                           ALXS0100
      EQUIVALENCE (C(3315),N)                                           ALXS0110
      DO 50 I = 2, N                                                    ALXS0170
      J = IPL(I-1)                                                      ALXS0200
   50 C(J+3) = VAR(I)                                                   ALXS0210
      T = VAR(1)                                                        ALXS0220
      DO 1 I=1,NOMOD                                                    ALXS0230
      L = MODNO(I)                                                      ALXS0240
      GO TO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,  ALXS0250
     123,24,25,26,27,28,29,30,31,32,33,34,35,36,37),L                   ALXS0260
    2 CALL A1                                                           ALXS0270
      GO TO 1                                                           ALXS0300
```

```
    3 CALL A2                                                           ALXS0310
      GO TO 1                                                           ALXS0320
    4 CALL A3                                                           ALXS0330
      GO TO 1                                                           ALXS0340
    5 CALL A4                                                           ALXS0350
      GO TO 1                                                           ALXS0360
    6 CALL A5                                                           ALXS0370
      GO TO 1                                                           ALXS0400
    7 CALL C1                                                           ALXS0410
      GO TO 1                                                           ALXS0420
    8 CALL C2                                                           ALXS0430
      GO TO 1                                                           ALXS0440
    9 CALL C3                                                           ALXS0450
      GO TO 1                                                           ALXS0460
   10 CALL C4                                                           ALXS0470
      GO TO 1                                                           ALXS0500
   11 CALL C5                                                           ALXS0510
      GO TO 1                                                           ALXS0520
   12 CALL C6                                                           ALXS0530
      GO TO 1                                                           ALXS0540
   13 CALL C7                                                           ALXS0550
      GO TO 1                                                           ALXS0560
   14 CALL C8                                                           ALXS0570
      GO TO 1                                                           ALXS0600
   15 CALL C9                                                           ALXS0610
      GO TO 1                                                           ALXS0620
   16 CALL C10                                                          ALXS0630
      GO TO 1                                                           ALXS0640
   17 CALL D1                                                           ALXS0650
      GO TO 1                                                           ALXS0660
   18 CALL D2                                                           ALXS0670
      GO TO 1                                                           ALXS0700
   19 CALL D3                                                           ALXS0710
      GO TO 1                                                           ALXS0720
   20 CALL D4                                                           ALXS0730
      GO TO 1                                                           ALXS0740
   21 CALL D5                                                           ALXS0750
      GO TO 1                                                           ALXS0760
   22 CALL G1                                                           ALXS0770
      GO TO 1                                                           ALXS1000
   23 CALL G2                                                           ALXS1010
      GO TO 1                                                           ALXS1020
   24 CALL G3                                                           ALXS1030
      GO TO 1                                                           ALXS1040
   25 CALL G4                                                           ALXS1050
      GO TO 1                                                           ALXS1060
   26 CALL G5                                                           ALXS1070
      GO TO 1                                                           ALXS1100
```

```
   27 CALL G6                                                           ALXS1110
      GO TO 1                                                           ALXS1120
   28 CALL S1                                                           ALXS1130
      GO TO 1                                                           ALXS1140
   29 CALL S2                                                           ALXS1150
      GO TO 1                                                           ALXS1160
   30 CALL S3                                                           ALXS1170
      GO TO 1                                                           ALXS1200
   31 CALL S4                                                           ALXS1210
      GO TO 1                                                           ALXS1220
   32 CALL S5                                                           ALXS1230
      GO TO 1                                                           ALXS1240
   33 CALL S6                                                           ALXS1250
      GO TO 1                                                           ALXS1260
   34 CALL S7                                                           ALXS1270
      GO TO 1                                                           ALXS1300
   35 CALL S8                                                           ALXS1310
      GO TO 1                                                           ALXS1320
   36 CALL S9                                                           ALXS1330
      GO TO 1                                                           ALXS1340
   37 CALL S10                                                          ALXS1350
    1 CONTINUE                                                          ALXS1360
      DO 60 I = 2, N                                                    ALXS1370
      J = IFL(I-1)                                                      ALXS1400
   60 DER(I) = C(J)                                                     ALXS1410
      RETURN                                                            ALXS1420
      END                                                               ALXS1430
C
C
C
      SUBROUTINE AMRK                                                   AMRK0010
C                                                                       AMRK0020
C          DOUBLE PRECISION VERSION. SAVES FUNCTION VALUES AND          AMRK0030
C              THE INDEPENDANT VARIABLE IN FULL D.P.                    AMRK0040
C                                                                       AMRK0050
      DIMENSION V(1),D(1),EU(1),FL(1)                                   AMRK0170
      DOUBLE PRECISION P1,P2,P3,P4,C2,C3,C4                             AMRK0210
      DOUBLE PRECISION T(1500), TME, DELT
      COMMON C(3416) ,T
      EQUIVALENCE  (C(3014),V(1))
      EQUIVALENCE  (C(2913) , D(1)   )
      EQUIVALENCE  (C(3215) , EU(1)  )
      EQUIVALENCE  (C(3115) , FL(1)  )
      EQUIVALENCE (C(2659), ONCE)
      EQUIVALENCE (C(3315) , NI   )                                     AMRK0110
      EQUIVALENCE (C(2912) , HMAX)                                      AMRK0140
      EQUIVALENCE (C(2911) , HMIN)                                      AMRK0150
      EQUIVALENCE (J1,NI)                                               AMRK0160
```

```
      IF (CNCE) 2,1,2                                                   AMRK0220
    1 CNCE = 1                                                          AMRK0230
      N = NI - 1                                                        AMRK0240
      CPT = 0.0                                                         AMRK0250
      P1  = 2.2916666666666667                                          AMRK0260
      P2  = 2.4583333333333333                                          AMRK0270
      P3  = 1.5416666666666667                                          AMRK0300
      P4  = 0.375                                                       AMRK0310
      C2  = 0.7916666666666667                                          AMRK0320
      C3  = 0.2083333333333333                                          AMRK0330
      C4  = 0.0416666666666667                                          AMRK0340
      KOUNT = 0                                                         AMRK0350
    2 N1 = N                                                            AMRK0360
      J2=J1+N1                                                          AMRK0370
      J3=J2+N1                                                          AMRK0400
      J4=J3+N1                                                          AMRK0410
      J5=J4+N1                                                          AMRK0420
      J6=J5+N1                                                          AMRK0430
      J7=J6+N1                                                          AMRK0440
      J8=J7+N1                                                          AMRK0450
      J9=J8+N1                                                          AMRK0460
      J10=J9+N1                                                         AMRK0470
      J11=J10+N1                                                        AMRK0500
      J12=J11+N1                                                        AMRK0510
      J13=J12+N1                                                        AMRK0520
      J14=J13+N1                                                        AMRK0530
      J15=J14+N1                                                        AMRK0540
C     IF (CPT.NE 0.) GO TO 60                                           AMRK0550
      IF (OPT) 60,49,60                                                 AMRK0560
C     IF (DELT . EQ. (0.5*D(1))) GO TO 200                              AMRK0570
   49 IF (DELT - (0.5*D(1))) 50,200,50                                  AMRK0600
 50   KOUNT=0                                                           AMRK0610
C                                                                       AMRK0620
C     START RUNGE-KUTTA INTEGRATION.                                    AMRK0630
C     COMPUTE K0                                                        AMRK0640
C                                                                       AMRK0650
   60 CONTINUE                                                          AMRK0660
      DO 25 I=1,N1                                                      AMRK0670
      K5=J15+I                                                          AMRK0700
C     IF(SNGL(T(K5)) .NE. V(I+1))T(K5) = V(I+1)                         AMRK0710
      IF(SNGL(T(K5)) - V(I+1)) 24,25,24                                 AMRK0720
   24 T(K5) = V(I+1)                                                    AMRK0730
 25   CONTINUE                                                          AMRK0740
C     IF(SNGL(TME) .NE. V(1)) TME = V(1)                                AMRK0750
   26 TME=V(1)
   27 KOUNT = KOUNT + 1                                                 AMRK1000
   70 DO 80 I=1,N1                                                      AMRK1010
      K0=J7+I                                                           AMRK1020
```

```
   80 T(K0)=D(1)*D(I+1)                                                  AMRK1030
C                                                                        AMRK1040
C     COMPUTE K1                                                         AMRK1050
C                                                                        AMRK1060
      DELT=0.5*D(1)                                                      AMRK1070
      TME=DELT+TME                                                       AMRK1100
      V(1)=TME                                                           AMRK1110
      DO 90 I=1,N1                                                       AMRK1120
      K0=J7+I                                                            AMRK1130
      K1=J11+I                                                           AMRK1140
      K2=J12+I                                                           AMRK1150
      K3=J13+I                                                           AMRK1160
      K5=J15+I                                                           AMRK1170
      T(I)=D(I+1)                                                        AMRK1200
      T(K1)=T(K2)                                                        AMRK1210
      T(K2)=T(K3)                                                        AMRK1220
      T(K3)=T(K5)                                                        AMRK1230
      T(K5) =T(K5) +0.5*T(K0)                                            AMRK1240
 90   V(I+1)=T(K5)                                                       AMRK1250
      CALL ALXSLE                                                        AMRK1260
      DO 100 I=1,N1                                                      AMRK1270
      K1=J8+I                                                            AMRK1300
  100 T(K1)=D(1)*D(I+1)                                                  AMRK1310
C                                                                        AMRK1320
C     COMPUTE K2                                                         AMRK1330
C                                                                        AMRK1340
      DO 110 I=1,N1                                                      AMRK1350
      K5=J15+I                                                           AMRK1360
      K1=J8+I                                                            AMRK1370
      K0=J7+I                                                            AMRK1400
      T(K5) =T(K5) +0.5*(T(K1)-T(K0))                                    AMRK1410
 110  V(I+1)=T(K5)                                                       AMRK1420
      CALL ALXSLE                                                        AMRK1430
      DO 120 I=1,N1                                                      AMRK1440
      K2=J9+I                                                            AMRK1450
  120 T(K2)=D(1)*D(I+1)                                                  AMRK1460
C                                                                        AMRK1470
C     COMPUTE K3                                                         AMRK1500
C                                                                        AMRK1510
      TME=DELT+TME                                                       AMRK1520
      V(1)=TME                                                           AMRK1530
      DO 130 I=1,N1                                                      AMRK1540
      K5=J15+I                                                           AMRK1550
      K2=J9+I                                                            AMRK1560
      K1=J8+I                                                            AMRK1570
      T(K5) =T(K5) +T(K2)-0.5*T(K1)                                      AMRK1600
 130  V(I+1)=T(K5)                                                       AMRK1610
      CALL ALXSLE                                                        AMRK1620
```

```
      DO 140 I=1,N1                                                     AMRK1630
      K3=J10+I                                                          AMRK1640
  140 T(K3)=D(1)*D(I+1)                                                 AMRK1650
C                                                                       AMRK1660
C     COMPLTE VALUE OF FUNCTION.                                        AMRK1670
C                                                                       AMRK1700
      DO 150 I=1,N1                                                     AMRK1710
      K0=J7 +I                                                          AMRK1720
      K1=J8 +I                                                          AMRK1730
      K2=J9 +I                                                          AMRK1740
      K3=J10+I                                                          AMRK1750
      K5=J15+I                                                          AMRK1760
      T(K5)=T(K5)+0.1666666666666667*                                   AMRK1770
     1(T(K0)+T(K1)+T(K1)-4.*T(K2)+T(K3))                                AMRK2000
      V(I+1)=T(K5)                                                      AMRK2010
  150 CONTINUE                                                          AMRK2020
  170 CONTINUE                                                          AMRK2030
      CALL AUXSUB                                                       AMRK2040
      DO 180 I=1,N1                                                     AMRK2050
      K5=J1+I                                                           AMRK2060
      K0=J2+I                                                           AMRK2070
      K1=J3+I                                                           AMRK2100
      K2=J4+I                                                           AMRK2110
      K3=J5+I                                                           AMRK2120
      K4=J6+I                                                           AMRK2130
      T(K4)=T(K3)                                                       AMRK2140
      T(K3)=T(K2)                                                       AMRK2150
      T(K2)=T(K1)                                                       AMRK2160
      T(K1)=T(K0)                                                       AMRK2170
      T(K0)=T(K5)                                                       AMRK2200
      T(K5)=T(I)                                                        AMRK2210
       T(I)=D(I+1)                                                      AMRK2220
 180  CONTINUE                                                          AMRK2230
 190   CONTINUE                                                         AMRK2240
      RETURN                                                            AMRK2250
C                                                                       AMRK2260
C     DO ADAMS-MOULTON INTEGRATION                                      AMRK2270
C                                                                       AMRK2300
 200  CONTINUE                                                          AMRK2310
C     IF (KOUNT .LT. 3) GO TO 60                                        AMRK2320
      IF (KOUNT - 3) 60,201,201                                         AMRK2330
  201 KOUNT = KOUNT + 1                                                 AMRK2340
      DELT=D(1)*.5                                                      AMRK2350
      DO 210 I=1,N1                                                     AMRK2360
      K0=J7+I                                                           AMRK2370
      K1=J2+I                                                           AMRK2400
      K2=J3+I                                                           AMRK2410
      K4=J1+I                                                           AMRK2420
```

```
      K5=J15+I                                                          ANRK2430
C                                                                       ANRK2440
C     COMPUTE Y-PREDICTED.                                              ANRK2450
C                                                                       ANRK2460
      T(K0)=T(K5)                                                       ANRK2470
      T(K5) =T(K0)+D(1)*(P1*T(I)-P2*T(K4)+P3*T(K1)-P4*T(K2))            ANRK2500
  210 V(I+1)=T(K5)                                                      ANRK2510
      TME=TME+D(1)                                                      ANRK2520
      V(1)=TME                                                          ANRK2530
      CALL ALXSUB                                                       ANRK2540
      K5=0                                                              ANRK2550
      DO 220 I=1,N1                                                     ANRK2560
      K0=J8+I                                                           ANRK2570
      K2=J2+I                                                           ANRK2600
      K1=J7+I                                                           ANRK2610
      K4=J1+I                                                           ANRK2620
      K3=J15+I
C                                                                       ANRK2630
C     COMPUTE Y-CORRECTED                                               ANRK2640
C                                                                       ANRK2650
      T(K0)=T(K1) +D(1)*(P4*D(I+1)+C2*T(I)-C3*T(K4)+T(K2)*C4)           ANRK2660
      ERF=14.*DMAX1(DABS(T(K1)),.01D0)
      ERU=ERF*EL(I)
      ERL=ERF*EL(I)
      TEMP = DABS(T(K0)-T(K3))
C     IF TEMP.LT.ERU) GO TO 215
      IF (TEMP-ERU) 215,214,214
C     IF(ABS(SNGL(DELT)) .GE. HMIN) GO TO 260                           ANRK2720
  214 IF(ABS(SNGL(DELT)) - HMIN) 215,260,260                            ANRK2730
  215 CONTINUE                                                          ANRK2740
C     IF (TEMP.LT.ERL)  K5=K5+1
      IF (TEMP-ERL) 216,220,220
  216 K5 = K5 + 1                                                       ANRK2770
  220 CONTINUE                                                          ANRK3000
C     IF(K5 .LT. N1) GO TO 300                                          ANRK3010
      IF(K5 - N1) 300,221,221                                           ANRK3020
C     IF(ABS(D(1) + D(1)) .GT. HMAX) GO TO 300                          ANRK3030
  221 IF(ABS(D(1) + D(1)) - HMAX) 222,222,300                           ANRK3040
C                                                                       ANRK3050
C     SET-UP FOR DOUBLING STEP SIZE                                     ANRK3060
C                                                                       ANRK3070
C     IF(KOUNT .LE. 6) GO TO 300                                        ANRK3100
  222 IF(KOUNT - 6) 300,300,223                                         ANRK3110
  223 CONTINUE                                                          ANRK3120
      DO 240 I=1,N1                                                     ANRK3130
      K1=J1+I                                                           ANRK3140
       K2=J2+I                                                          ANRK3150
      K3=J3+I                                                           ANRK3160
```

```
      K5=J5+I                                                           AMRK3170
      T(I)=T(K1)                                                        AMRK3200
      T(K1)=T(K3)                                                       AMRK3210
  240 T(K2)=T(K5)                                                       AMRK3220
      D(1)=D(1)+D(1)                                                    AMRK3230
      KOUNT=4                                                           AMRK3240
      DELT=.5*D(1)                                                      AMRK3250
      GO TO 300                                                         AMRK3260
C                                                                       AMRK3270
C     SET-UP FOR HALVING STEP SIZE.                                     AMRK3300
C                                                                       AMRK3310
  260 CONTINUE                                                          AMRK3320
C     IF(KOUNT .LE. 4) GO TO 350                                        AMRK3330
      IF(KOUNT - 4) 350,350,261                                         AMRK3340
  261 TME=TME-D(1)                                                      AMRK3350
      V(1)=TME                                                          AMRK3360
      D(1)=DELT                                                         AMRK3370
      DELT=.5*D(1)                                                      AMRK3400
      DO 265 I=1,N1                                                     AMRK3410
      K0=J7+I                                                           AMRK3420
      K1=J1+I                                                           AMRK3430
      K2=J2+I                                                           AMRK3440
      K3=J3+I                                                           AMRK3450
      K5=J15+I                                                          AMRK3460
      T(K5)=T(K0)                                                       AMRK3470
      V(I+1)=T(K0)                                                      AMRK3500
      T(K3)=T(K2)+0.5*(T(K1)-T(K2))                                     AMRK3510
      T(K2)=T(K1)                                                       AMRK3520
  265 T(K1)=T(K1)+0.5*(T(I)-T(K1))                                      AMRK3530
      KOUNT=4                                                           AMRK3540
      GO TO 200                                                         AMRK3550
C                                                                       AMRK3560
C          INTEGRATION IS FINISHED. SET UP DERIVATIVES AND EXIT.        AMRK3570
C                                                                       AMRK3600
  300 DO 310 I=1,N1                                                     AMRK3610
      K5=J15+I                                                          AMRK3620
      K0=J8+I                                                           AMRK3630
      T(K5)=T(K0)                                                       AMRK3640
  310 V(I+1)=T(K0)                                                      AMRK3650
      GO TO 170                                                         AMRK3660
  350 CONTINUE                                                          AMRK3670
C                                                                       AMRK3700
C          RETURN TO 3RD RK INTEGRATION AND RESTART                     AMRK3710
C                                                                       AMRK3720
      DO 360 I=1,N1                                                     AMRK3730
      K5=J15+I                                                          AMRK3740
      K1=J11+I                                                          AMRK3750
      T(K5)=T(K1)                                                       AMRK3760
```

```
  360 V(I+1)=T(K1)                                                       AMRK3770
      TME=TME-C(1)*4.                                                    AMRK4000
      V(1)=TME                                                           AMRK4010
       D(1)=DELT                                                         AMRK4020
      CALL AUXSUB                                                        AMRK4030
      GO TO 50                                                           AMRK4040
      END                                                                AMRK4050
C
C
C
      SUBROUTINE PROCES
      RETURN
      END
C
C
C
      SUBROUTINE RESET
C
      COMMON C(3415)
      EQUIVALENCE (C(2300),NOLIST),(C(2301),LISTNO(1)),(C(2351),
     1 VALUE(1))
      EQUIVALENCE (C(2675),KRUN)
      DIMENSION LISTNO(50), VALUE(50)
      IF(NOLIST.EQ.0) RETURN
      DO 1 I=1,NOLIST
      J=LISTNO(I)
    1 C(J)=VALUE(I)
      KRUN=KRUN+1
      RETURN
      END
C
C
C
      SUBROUTINE INPT1                                                   INPT0010
C
      COMMON C(3415)
      COMMON/CPLS/ ENG(30),IENG(15)
      NAMELIST /MCNT/ ENG,IENG                                             F  96
      EQUIVALENCE (C(2675), KRUN)
      FLG1=0.0
      IF (KRUN.GT.0) GO TO 100
      READ (5,MCNT)                                                        F 106
      WRITE (6,MCNT)                                                       F 108
  100 RETURN
      END
C
C
      SUBROUTINE INPT2                                                   INPT0010
```

```
      RETURN                                                            INPT0020
      END                                                               INPT0030
C
C
      SUBROUTINE INPT3                                                  INPT0010
      RETURN                                                            INPT0020
      END                                                               INPT0030
C
C
      SUBROUTINE OUPT1                                                  OUPT0010
      RETURN                                                            OUPT0020
      END                                                               OUPT0030
C
C
C
      SUBROUTINE OUPT2                                                  OUTP0020
C
C     OUTPUT INITIALIZATION SUBROUTINE OUPT2                            OUTP0010
C
      COMMON C(3415)                                                    OUTP0030
      INTEGER PGCNT,DTCNT                                               OUTP0130
      EQUIVALENCE   (C(2660),DTCNT)                                     OUTP0040
      EQUIVALENCE   (C(2661),NCOUT)                                     OUTP0050
      EQUIVALENCE   (C(2662),PGCNT)                                     OUTP0060
      EQUIVALENCE (C(2667),ITCNT),(C(2668),PCNT),(C(2669),CPP)          OUTP0070
      EQUIVALENCE (C(2670), TAPE), (C(2671), TAPEND)                    OUTP0100
      EQUIVALENCE (C(2909),DCC)                                         OUTP0110
      EQUIVALENCE(C(2910),KCONV)                                        OUTP0120
      KCONV=0                                                           OUTP0140
      ITCNT = DCC + 1.0                                                 OUTP0150
      PCNT = 0.0                                                        OUTP0160
      PGCNT = 1                                                         OUTP0170
      DTCNT = (NCOUT + 4)/5                                             OUTP0200
C      IF (ITCNT .GE. 7) GO TO 2                                        OUTP0210
      IF (ITCNT - 7) 1,2,2                                              OUTP0220
    1 WRITE (6,6) (I,C(I),C(I+1),C(I+2),C(I+3),C(I+4),C(I+5),C(I+6),    OUTP0230
     1  C(I+7),I=1,3415,8)                                              OUTP0240
    6 FORMAT(1H1/(I5,2X,8E14.7))                                        OUTP0250
    2 TAPEND = 0.                                                       OUTP0260
      K = TAPE                                                          OUTP0270
C      IF (K. NE. 0) REWIND K                                           OUTP0300
      IF (K) 3,4,3                                                      OUTP0310
    3 REWIND K                                                          OUTP0320
    4 RETURN                                                            OUTP0330
      END                                                               OUTP0340
C
C
C
```

```
      SUBROUTINE CLPT3                                                  CLPT0020
C                                                                       
C     OUTPLT SUBROUTINE CLPT3                                           CLPT0010
C                                                                       
      DIMENSION B(50),OUTNO(50),ONAME1(50),ONAME2(50)                   CLPT0170
      DIMENSION ONAME3(50)
      COMMON C(3415)                                                    CLPT0030
      INTEGER ITCNT,PGCNT,OUTNO                                         CLPT0290
      EQUIVALENCE (C(2500),OUTNO(1))
      EQUIVALENCE (C(2550),ONAME1(1))
      EQUIVALENCE (C(2600),ONAME2(1))
      EQUIVALENCE (C(2150) , ONAME3(1))
      EQUIVALENCE    (C(2660),DTCNT)                                    CLPT0070
      EQUIVALENCE    (C(2661),NOOUT)                                    CLPT0100
      EQUIVALENCE    (C(2662),PGCNT)                                    CLPT0110
      EQUIVALENCE (C(2667),ITCNT),(C(2668),PCNT),(C(2669),CPP)          CLPT0120
      EQUIVALENCE    (C(932), T)                                        CLPT0130
      EQUIVALENCE (C(2913), DER)                                        CLPT0140
      EQUIVALENCE (C(2670), TAPE), (C(2671), TAPEND)                    CLPT0150
      EQUIVALENCE (C(0290),XTP),(C(0298),YTP),(C(0306),ZTP)             CLPT0160
      EQUIVALENCE (C(3400),PUNCH)
      EQUIVALENCE (C(0521),THET)
      EQUIVALENCE (C(0522),PSI)
      EQUIVALENCE (C(0523),PHI)
      EQUIVALENCE (C(0554),AXECI)
      EQUIVALENCE (C(0555),AYECI)
      EQUIVALENCE (C(0556),AZECI)
      EQUIVALENCE (C(0577),VXECI)
      EQUIVALENCE (C(0552),VYECI)
      EQUIVALENCE (C(0553),VZECI)
      EQUIVALENCE (C(0547),XECI)
      EQUIVALENCE (C(0548),YECI)
      EQUIVALENCE (C(0549),ZECI)
C     IF (ITCNT .GT. 6) GO TO 7                                         CLPT0210
      IF (ITCNT - 6) 10,10,7                                            CLPT0220
   10 ITCNT = ITCNT + 1                                                 CLPT0230
      WRITE (6,6) (I,C(I),C(I+1),C(I+2),C(I+3),C(I+4),C(I+5),C(I+6),    CLPT0240
     1 C(I+7),I=1,3415,8)                                               CLPT0250
    6 FORMAT(1H1/(I5,2X,8E14.7))                                        CLPT0260
      PGCNT = 1                                                         CLPT0270
    7 CONTINUE
C   8 IF (T .LT. PCNT) RETURN                                           CLPT0340
    8 IF (T - PCNT) 12,9,9                                              CLPT0350
   12 RETURN                                                            CLPT0360
    9 PCNT=PCNT+CPP
C     IF (PGCNT .NE. 1) GO TO 3                                         CLPT0400
      IF (PGCNT - 1) 3,1,3                                              CLPT0410
    1 WRITE(6,2) (ONAME1(I),ONAME2(I),ONAME3(I), I = 1, NOOUT)
```

```
      2 FORMAT (1H1,5X,4HTIME,5X,5(8X,3A4)//(23X,3A4,8X,3A4,8X,3A4,8X,
       1 3A4,8X,3A4)/)
        PGCNT = 2*DTCNT + 4                                             CLPT0450
C     3 IF (PGCNT .GE. 58) GO TO 1                                      CLPT0460
      3 IF (PGCNT - 58) 13,1,1                                          CLPT0470
     13 DO 4 I =1, NOOUT                                                CLPT0500
        J = OUTNC(I)                                                    CLPT0510
      4 B(I) = C(J)                                                     CLPT0520
        K = TAPE                                                        CLPT0530
C       IF (K .NE. 0) WRITE (K) T,XTP,YTP,ZTP,TAPENC, (B(I),I=1,NOOUT)  CLPT0540
        IF (K) 14,15,14                                                 CLPT0550
     14 WRITE (K) T,XTP,YTP,ZTP,TAPEND, (B(I),I=1,NOOUT)                CLPT0560
     15 WRITE (6,5) T,(B(I),I=1,NOOUT)                                  CLPT0570
      5 FORMAT (////2X,E15.8,5E20.8/(17X,5E20.8))                       CLPT0600
   22   FORMAT(F5.1,3E10.4,3E15.9)
        IF(PUNCH)20,20,21
    21  WRITE(17,22) T,AXECI,AYECI,AZECI,XECI,YECI,ZECI
        WRITE(17,22) T,VXECI,VYECI,VZECI,THET,PSI,PHI
    20  CONTINUE
        PGCNT = PGCNT + DTCNT + 4                                       CLPT0610
        RETURN                                                          CLPT0620
        END                                                             CLPT0630
C
C
        SUBROUTINE STGE1                                                STGE0010
        RETURN                                                          STGE0020
        END                                                             STGE0030
C
C
C
        SUBROUTINE STGE2                                                STGE0010
C
        COMMON C(3415)                                                  STGE0020
        EQUIVALENCE (C(2904),KSTEP)                                     STGE0030
        EQUIVALENCE (C(2910), KCONV)                                    STGE0040
        EQUIVALENCE  (C(2672) , LCONV        )                          STGE0050
        KCONV = 0                                                       STGE0060
        LCONV = 0                                                       STGE0070
        KSTEP = 1                                                       STGE0100
        RETURN                                                          STGE0110
        END                                                             STGE0120
C
C
C
        SUBROUTINE STGE3                                                STGE0140
C
C       THIS OPERATIONAL SUBROUTINE STAGES WHEN IMPACT IS MADE WITH THE STGE0010
C       EARTH, WHEN FINAL TIME TF IS REACHED, OR WHEN ANOTHER PART OF THE STGE0020
```

```
C     PROGRAM SETS LCONV = 2.  STAGING WILL OCCUR WHEN THE VEHICLE IS    STGE0030
C     WITHIN + OR - ONE FOOT OF THE EARTH SURFACE, WHEN T IS WITHIN + ORSTGE0040
C     - 0.0001 SECOND OF TF, OR ACCORDING TO SOME CRITERION IN ANOTHER  STGE0050
C     MODULE OR SUBROUTINE.  FOR EXAMPLE, IF STAGING IS DESIRED AT      STGE0060
C     COINCIDENCE BETWEEN A AND B, THE FOLLOWING FORTRAN STATEMENTS     STGE0070
C     SHOULD BE USED IN ONE OF THE SUBROUTINES OF MODULES --            STGE0100
C                                                                       STGE0110
C     IF(ABS(A-B) .LE. 0.5) LCONV = 2                                   STGE0120
C     IF( (B-A) .LT. 0.) LCONV = 1                                      STGE0130
      DIMENSION STATNO(100),RMS(10)                                     STGE0270
      COMMON C(3415)                                                    STGE0150
      INTEGER STATNO                                                    STGE0300
      EQUIVALENCE  (C(932),T),(C(2913),DER)                             STGE0160
      EQUIVALENCE (C(2668),PCNT)                                        STGE0170
      EQUIVALENCE (C(0550),TF), (C(0507),H)                             STGE0200
      EQUIVALENCE (C(2904),KSTEP), (C(2905),STEP)                       STGE0210
      EQUIVALENCE (C(2670), TAPE), (C(2671), TAPEND)                    STGE0220
      EQUIVALENCE (C(2441),NOSTAT),(C(2200),STATNO(1)),(C(2430),RMS(1))
      EQUIVALENCE (C(2442),LOSTAT)                                      STGE0240
      EQUIVALENCE (C(2910), KCONV)                                      STGE0250
      EQUIVALENCE  (C(2672) , LCONV     )                               STGE0260
C     IF (ABS(H) .LE. 1.0) GO TO 2                                      STGE0310
      IF (ABS(H)-1.0) 2,2,5                                             STGE0320
C     IF (H .LT. 0.) GO TO 1                                            STGE0330
    5 IF (H) 1,6,6                                                      STGE0340
C     IF (ABS(T-TF) .LE. 0.0001) GO TO 2                                STGE0350
    6 IF (ABS(T-TF) - 0.0001) 2,2,7                                     STGE0360
C     IF ( (TF-T) .LT. 0.) GO TO 1                                      STGE0370
    7 IF (TF-T) 1,8,8                                                   STGE0400
C     IF (LCONV .EQ. 2) GO TO 2                                         STGE0410
    8 IF (LCONV - 2) 9,2,9                                              STGE0420
C     IF (LCONV .EQ. 1) GO TO 1                                         STGE0430
    9 IF (LCONV - 1) 10,1,10                                            STGE0440
C     IF (DER .LT. 0.) DER = -DER*0.5                                   STGE0450
   10 IF (DER) 11,12,12                                                 STGE0460
   11 DER = -DER*0.5                                                    STGE0470
   12 RETURN                                                            STGE0500
C   1 IF (DER .GT. 0.) DER = -DER*0.5                                   STGE0510
    1 IF (DER) 14,14,13                                                 STGE0520
   13 DER = -DER*0.5                                                    STGE0530
   14 KCONV = KCONV + 1                                                 STGE0540
C     IF (KCONV .GE. 10) GO TO 2                                        STGE0550
      IF (KCONV - 10) 15,2,2                                            STGE0560
   15 RETURN                                                            STGE0570
    2 PCNT = 1.0                                                        STGE0600
      TAPEND = 1.                                                       STGE0610
      CALL CLFT3                                                        STGE0620
      K = TAPE                                                          STGE0630
```

```
C     IF (K .NE. 0) END FILE K                                          STGE0640
C     IF (K .NE. 0) REWIND K                                            STGE0650
      IF (K) 16,17,16                                                   STGE0660
   16 END FILE K                                                        STGE0670
      REWIND K                                                          STGE0700
   17 KSTEP = 2                                                         STGE0710
C     IF (LCSTAT .EQ. NOSTAT) GO TO 4                                   STGE0720
      IF (LCSTAT - NOSTAT) 18,4,18                                      STGE0730
   18 L = LCSTAT + 1                                                    STGE0740
      DO 3 I = L, NOSTAT                                                STGE0750
      M = STATNO(I)                                                     STGE0760
    3 RMS(I) = C(M)                                                     STGE0770
    4 RETURN                                                            STGE1000
      END                                                               STGE1010
C
C
      SUBROUTINE CNTR1                                                  CNTR0010
      RETURN                                                            CNTR0020
      END                                                               CNTR0030
C
C
      SUBROUTINE CNTR2                                                  CNTR0010
      RETURN                                                            CNTR0020
      END                                                               CNTR0030
C
C
      SUBROUTINE CNTR3                                                  CNTR0010
      RETURN                                                            CNTR0020
      END                                                               CNTR0030
C
C
      SUBROUTINE RNDM1                                                  RNDM0010
      RETURN                                                            RNDM0020
      END                                                               RNDM0030
C
C
C
      SUBROUTINE RNDM2
C
      COMMON C(3415)
      EQUIVALENCE (C(2443), RNFLG)
      EQUIVALENCE (C(2446), ZN)
      EQUIVALENCE (C(2448), NOFNDM)
      EQUIVALENCE (C(2449), DELT)
      EQUIVALENCE (C(2450), RNDMNO)
      EQUIVALENCE (C(2913), CEF)
      DIMENSION RNDMNO(50)
      INTEGER RNDMNO
```

```
      IF (NCRNDM) 2,2,3
    3 RNFLG=0.
      DELT=DER
      DO 1 I=1,NCRNDM
      J=RNDMNO(I)
      C(J+5)=2.718281828**(-DER*C(J+4))
      C(J+6)=C(J+3)*SQRT(1.0-C(J+5)*C(J+5))
      ZN=C(J+1)
      Y=C(J)
      N=INT(Y)
      CALL RFG(N,X)
      C(J+7)=C(J+3)*X
    1 C(J+1)=ZN
    2 RETURN
      END
C
C
C
      SUBROUTINE RNDM3
C
      COMMON C(3415)
      EQUIVALENCE (C(2443), RNFLG)
      EQUIVALENCE (C(2444), RN)
      EQUIVALENCE (C(2446), ZN)
      EQUIVALENCE (C(2448), NCRNDM)
      EQUIVALENCE (C(2449), DELT)
      EQUIVALENCE (C(2450), RNDMNO)
      EQUIVALENCE (C(2913), DER)
      DIMENSION RNDMNO(50)
      INTEGER RNDMNO
      IF (NCRNDM) 20,20,10
   10 IF (RNFLG.GE.1.0) GO TO 8
      IF (DER.GE.0.) GO TO 6
    8 DO 9 I=1,NCRNDM
      J=RNDMNO(I)
      X=(C(J+7)-C(J+5)*C(J+8))/C(J+6)
      IF (RNFLG.GE.1.0)  GO TO 5
      RNFLG=1.0
      DELT=-DER
      GO TO 7
    5 DELT=DELT+DER
    7 C(J+5)=2.718281828**(-DELT*C(J+4))
      C(J+6)=C(J+3)*SQRT(1.-C(J+5)*C(J+5))
    9 C(J+7)=C(J+6)*X+C(J+5)*C(J+8)
      RETURN
    6 IF (DELT.EQ.DER) GO TO 4
    2 DO 3 I=1,NCRNDM
      J=RNDMNO(I)
```

```
      C(J+5)=2.718281828**(-DER*C(J+4))
    3 C(J+6)=C(J+3)*SQRT(1.0-C(J+5)*C(J+5))
      DELT=DER
    4 DO 1 I=1,NCRNDM
      J=RNDMNC(I)
      Y=C(J)
      ZN=C(J+1)
      N=INT(Y)
      CALL RFG(N,X)
      C(J+8)=C(J+7)
      C(J+7)=C(J+6)*X+C(J+5)*C(J+7)
    1 C(J+1)=ZN
   20 RETURN
        END
C
C
      SUBROUTINE AUXA1                                                  AUXA0010
      RETURN                                                            AUXA0020
      END                                                               AUXA0030
C
C
      SUBROUTINE AUXA2
      RETURN
      END
C
C
      SUBROUTINE AUXA3
      RETURN
      END
C
C
      SUBROUTINE AUXB1                                                  AUXB0010
      RETURN                                                            AUXB0020
      END                                                               AUXB0030
C
C
      SUBROUTINE AUXB2                                                  AUXB0010
      RETURN                                                            AUXB0020
      END                                                               AUXB0030
C
C
      SUBROUTINE AUXB3                                                  AUXB0010
      RETURN                                                            AUXB0020
      END                                                               AUXB0030
C
C
      SUBROUTINE AUXC1                                                  AUXC0010
      RETURN                                                            AUXC0020
```

```
      END                                                               ALXC0030
C
C
      SUBROUTINE ALXC2                                                  ALXC0010
      RETURN                                                            ALXC0020
      END                                                               ALXC0030
C
C
      SUBROUTINE AUXC3                                                  ALXC0010
      RETURN                                                            ALXC0020
      END                                                               ALXC0030
C
C
C
      SUBROUTINE RFG(N,X)
C
      COMMON C(3415)
      EQUIVALENCE (C(2446), ZN)
      Y=FLOAT(N)
      X=0.0
C     GENERATE UNIFORM RANDOM NO.(0 TO 1)
C    SUM FOR RAND NO(SIGMA=1,MEAN=0)
C     UNIFORM DIST FOR N=1
C     NORMAL DIST FOR LARGE N (12 OR GREATER)
      DO 1 I=1,N
      X=X+ZN-0.5
      RNO=997.*ZN
    1 ZN=RNO-AINT(RNO)
      X=SQRT(12./Y)*X
      RETURN
      END
C
C
C
      SUBROUTINE A1I
C
C     AERODYNAMIC FORCES AND MOMENTS INITIALIZATION MODULE A1I
C     BODY AXES
C
      COMMON C(3415)
      EQUIVALENCE  (C(0208) , TSA  )
      EQUIVALENCE  (C(0257) , CTS  )
      EQUIVALENCE  (C(0258) , STS  )
      CTS = COS(TSA)
      STS = SIN(TSA)
      RETURN
      END
C
```

```
C
C
      SUBROUTINE A1
C
C     AERODYNAMIC FORCES AND MOMENTS MODULE A1 - BODY AXES
C
      COMMON C(3415)
      EQUIVALENCE (C(0110), S)
      EQUIVALENCE (C(0111), CA)
      EQUIVALENCE (C(0113), CY)
      EQUIVALENCE (C(0115), CZ)
      EQUIVALENCE (C(0116), CBAR)
      EQUIVALENCE (C(0118), CMQ)
      EQUIVALENCE (C(0119), CNR)
      EQUIVALENCE (C(0120), CLP)
      EQUIVALENCE (C(0121), CM)
      EQUIVALENCE (C(0122), CN)
      EQUIVALENCE (C(0123), CL)
      EQUIVALENCE (C(0128), FXBA)
      EQUIVALENCE (C(0129), FYBA)
      EQUIVALENCE (C(0130), FZBA)
      EQUIVALENCE (C(0131), CLBA)
      EQUIVALENCE (C(0132), CMBA)
      EQUIVALENCE (C(0133), CNBA)
      EQUIVALENCE (C(0212), PBA)
      EQUIVALENCE (C(0216), QBA)
      EQUIVALENCE (C(0220), RBA)
      EQUIVALENCE (C(0257), CTS)
      EQUIVALENCE (C(0258), STS)
      EQUIVALENCE (C(0508), CD)
      EQUIVALENCE (C(0529), VAT)
      CDS=CD*S
      CDSC=CDS*CBAR
      DIM=CBAR/(2.0*VAT)
C    AERODYNAMIC MOMENTS (BODY AXES)
      CLBA=CDSC*(CL+CLP*DIM*PBA)
      CMBA=CDSC*(CM+CMQ*DIM*QBA)
      CNBA=CDSC*(CN+CNR*DIM*RBA)
C    AERODYNAMIC FORCES (BODY AXES)
      FXBA=CDS*CA
      FYBA=CDS*CY
      FZBA=CDS*CZ
      RETURN
      END
C
C
      SUBROUTINE A2I
      RETURN
```

```
      END
C
C
C
      SUBROUTINE A2
C
C   CAB - ZERO-LIFT BODY DRAG
C   CAC - DRAG DUE TO DEFLECTED CONTROL SURFACES
C   CA - CAB+CAC
C   CZB - NORMAL FORCE DUE TO ANGLE OF ATTACK
C   CZC - NORMAL FORCE DUE TO FIN DEFLECTION
C   CZ - NORMAL FORCE
C   CYB - SIDE FORCE DUE TO SIDESLIP
C   CYC - SIDE FORCE DUE TO FIN DEFLECTION
C   CY - SIDE FORCE
C   CMQ - PITCH DAMPING MOMENT
C   CNR - YAW DAMPING MOMENT
C   CLP - ROLL DAMPING MOMT
C   CMB - PITCH MOMENT DUE TO ANGLE OF ATTACK
C   CMC - CONTROL SURFACE PITCH MOMENT
C   CM - CMB+CMC
C   CNB - YAW MOMENT DUE TO SIDESLIP
C   CNC - CONTROL SURFACE YAW MOMENT
C   CN  - CNB+CNC
C   CLB - INDUCED ROLL MOMENT
C   CLC - CONTROL SURFACE ROLL MOMENT
C   CL  - CLB+CLC
C
      COMMON/APPLE/TM1(7),TM2(15),TM3(14),TM4(5),TETA1(7),TETA2(7),
     1TETA3(6),TXI1(2),TXI2(10),TXI3(5),TDELT(4),TH1(3),TH2(2),
     2TCZBC(7,2,7),TDCZP(14,7),TCYBC(7,10,6),TCMBC(7,5,7),TDCMF(14,7),
     3TCDB(15,2),TCDC(15,2),TCAD2(15),TCNBC(7,10,6),TCMD1(15,4),
     4TCMD2(5,4),TCCMD(15),TDCMCA(15),TPMD(14,3),TCMQ(15),TPQ(14,3),
     5TCLBC(7,10,6),TCLD(15),TPLD(14,3),TCLP(15),TPP(14,3),TDCYSP(3),
     LTDCZSF(3),TDCMSP(3),TDCNSP(3),TH3(3),TFEJ(13),TTEJ(13)
      COMMON C(3415)
      EQUIVALENCE (C(0016), DP)
      EQUIVALENCE (C(0020),DQ)
      EQUIVALENCE (C(0024),DR)
      EQUIVALENCE (C(0111),CA)
      EQUIVALENCE (C(0113),CY)
      EQUIVALENCE (C(0114), XI)
      EQUIVALENCE (C(0115),CZ)
      EQUIVALENCE (C(0116),CBAR)
      EQUIVALENCE (C(0117),ETA)
      EQUIVALENCE (C(0118),CMQ)
      EQUIVALENCE (C(0119),CNR)
      EQUIVALENCE (C(0120),CLP)
```

```
EQUIVALENCE (C(0121),CM )
EQUIVALENCE (C(0122), CN)
EQUIVALENCE (C(0123), CL)
EQUIVALENCE (C(0136), CG)
EQUIVALENCE (C(0259), CGR)
EQUIVALENCE (C(0330), ALF)
EQUIVALENCE (C(0331), BET)
EQUIVALENCE (C(0332), DELQ)
EQUIVALENCE (C(0333), DELR)
EQUIVALENCE (C(0334), DELP)
EQUIVALENCE (C(0507), F)
EQUIVALENCE (C(0509), UBA)
EQUIVALENCE (C(0510), VBA)
EQUIVALENCE (C(0511), WBA)
EQUIVALENCE (C(0518), ASA)
EQUIVALENCE (C(0519), BSA)
EQUIVALENCE (C(0520), AM)
EQUIVALENCE (C(0529), VAT)
EQUIVALENCE (C(0932), T)
EQUIVALENCE (C(0933), T1)
EQUIVALENCE (C(0935), T3)
EQUIVALENCE (C(0936), T4)
EQUIVALENCE (C(1696), RDUL)
EQUIVALENCE (C(1700), RDUR)
EQUIVALENCE (C(1704), RDLL)
EQUIVALENCE (C(1708), RDLR)
EQUIVALENCE (C(0109), FLGSEP)
EQUIVALENCE (C(0306), ZTP)
EQUIVALENCE (C(0523), PHI)
RAD=57.2957795
ALF=ASA*RAD
BET=BSA*RAD
DUL=RDUL*RAD
DUR=RDUR*RAD
DLL=RDLL*RAD
DLR=RDLR*RAD
DELQ=DQ*RAD
DELR=DR*RAD
DELP=DP*RAD
ADUL=ABS(DUL)
ADUR=ABS(DUR)
ADLL=ABS(DLL)
ADLR=ABS(DLR)
COSETA=UBA/VAT
IF (COSETA.GE.1.0)  COSETA=1.0
IF (COSETA.GE.1.0)  COSETA=1.0
ETAR =ARCOS(COSETA)
ETA=ETAR*RAD
```

```
      VN=SQRT(VBA**2+WBA**2)
      IF (VN.LT.0.00001) GO TO 10
      XIR=ATAN2(VBA,WBA)
      GO TO 20
   10 XIR=0.0
   20 XI=XIR*RAD
      XI1=XI
      IF (XI.LT.0.0) XI1=-XI
      XI2=XI1
      IF (XI1.GE.90.) XI2=180.-XI1
C    TABLE LOOK-UP,ALL PHASES
      DCZP=0.0
      DCMP=0.0
      PMD=1.0
      PND=1.0
      PLD=1.0
      PQ=1.0
      PR=1.0
      PP=1.0
      DCMSEP=0.
      DCNSEP=0.
      DCLSEP=0.
      DCYSEP=0.
      DCZSEP=0.
      DCLSP=0.
      DCNSP=0.
      DCYSP=0.
C     MOMENT COEFFICIENTS
C      PITCHING MOMENT
      CMBCC=THREDL(AM,XI2,ETA,TM1,TXI3,TETA1,TCMBC,7,5,7)
      DCMUL=DDLI(AM,TM2,TDCMD,15)
      DCMUR=DCMLL
      DCMLL=-DCMUL
      DCMLR=-DCMUL
      DCMULA=DDLI(AM,TM2,TDCMDA,15)
      DCMURA=-DCMULA
      DCMLLA=-DCMULA
      DCMLRA=DCMLLA
      CMC=DDLI(AM,TM2,TCMC,15)
C      YAWING MOMENT
      CNBC=THREDL(AM,XI2,ETA,TM1,TXI2,TETA3,TCNBC,7,10,6)
      IF (XI1.GT.90.)  CNBC=-CNBC
      IF (XI.LT.0.0)  CNBC=-CNBC
      DCNUL=-DCMUL
      DCNUR=DCMUL
      DCNLL=-DCMLL
      DCNLR=DCMUL
      DCNULB=DCMLLA
```

AD-A085 147 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 16/2

GUIDANCE AND CONTROL OF TACTICAL MISSILES.(U)

DEC 79 T A GROTE

UNCLASSIFIED NL

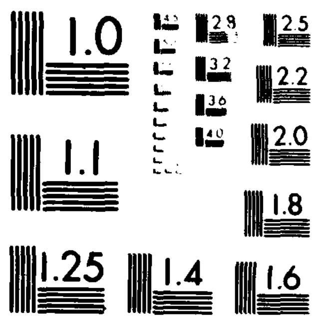
2 of 2

END

DATE FILMED

7 80

DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
      DCNURB=-DCMULA
      DCNLLB=-DCMULA
      DCNLRB=DCMULA
      CNR=CMC
C     ROLLING MOMENT
      CLBC=THREDL(AM,XI2,ETA,TM1,TXI2,TETA3,TCLBC,7,10,6)
      IF (XI1.GT.90.)  CLBC=-CLBC
      IF (XI.LT.0.)  CLBC=-CLBC
      CLD=CDLI(AM,TM2,TCLD,15)
      CLF=CDLI(AM,TM2,TCLF,15)
C    FORCE COEFFICIENTS
      CZBCC=THREDL(AM,XI2,ETA,TM1,TXI1,TETA1,TCZBC,7,2,7)
      CYBC=THREDL(AM,XI2,ETA,TM1,TXI2,TETA3,TCYBC,7,10,6)
      IF (XI1.GT.90.)  CYBC=-CYBC
      IF (XI.LT.0.)  CYBC=-CYBC
      CAD2=CDLI(AM,TM2,TCAD2,15)
      IF (T.GE.T3)  GO TO 100
C   TABLE LOOK-UP, SEP AND BOOST PHASE
C    ZERO-LIFT DRAG
      CDO=STDLIA(TM2,TH2,TCDB,AM,H,15,2)
      IF (T.LT.T1)  CDO=CDO+.08
C    CONTROL SURFACE EFFECTIVENESS
      CMULO=STDLIA(TM2,TDELT,TCMD1,AM,ADUL,15,4)
      IF (DUL.LT.0.0)  CMULO=-CMULO
      CMURO=STDLIA(TM2,TDELT,TCMD1,AM,ADUR,15,4)
      IF (DUR.GT.0.0)  CMURO=-CMURO
      CMLLO=STDLIA(TM2,TDELT,TCMD1,AM,ADLL,15,4)
      IF (DLL.LT.0.0)  CMLLO=-CMLLO
      CMLRO=STDLIA(TM2,TDELT,TCMD1,AM,ADLR,15,4)
      IF (DLR.GT.0.0)  CMLRO=-CMLRO
C    PLUME EFFECTS
      IF (AM.GE.1.6)  GO TO 200
      HOK=H/1000.
      FM=2.1212E-06*HOK**3-5.2017E-04*HOK**2+.044193*HOK-9.0905E-04
      DCMP=STDLIA(TM3,TETA2,TDCMP,AM,ETA,14,7)
      IF (DCMP.LT.0.0)  DCMP=0.0
      DCZP=STDLIA(TM3,TETA2,TDCZP,AM,ETA,14,7)
      IF (DCZP.GT.0.0)  DCZP=0.0
      DCMP=DCMP*FM
      DCZP=DCZP*FM
      PMD=STDLIA(TM3,TH1,TPMD,AM,H,14,3)
      IF (PMD.GT.1.0)  PMD=1.0
      PND=PMD
      PLD=STDLIA(TM3,TH1,TPLD,AM,H,14,3)
      IF (PLD.GT.1.0)  PLD=1.0
      PQ=STDLIA(TM3,TH1,TPQ,AM,H,14,3)
      IF (PQ.GT.1.0)  PQ=1.0
      PR=PQ
```

```
      PP=STCLIA(TM3,TH1,TFF,AM,H,14,3)
      IF (PP.GT.1.0)  PP=1.C
C     SEPARATICN EFFECTS
      IF (FLGSEP.LE.0.) GC TC 200
      IF (T.GE.1.0) GO TO 200
      IF (ZTF.GE.15.) GC TC 200
      FS2=(1.-ZTF/15.)*EXP(-ZTF/15.)
      DCMSPP=CDLI(H,TH3,TCCMSP,3)
      DCZSPP=CDLI(H,TH3,TCCZSP,3)
      DCMSP=FS2*DCMSPP
      DCZSP=FS2*DCZSPP
      IF (ZTP.GE.4.0) GO TC 30
      FS1=(1.-ZTP/4.)*EXP(-ZTP/4.)
      DCYSPP=CDLI(H,TH3,TCCYSP,3)
      DCNSPP=CDLI(H,TH3,TCCNSP,3)
      DCYSP=FS1*DCYSPP
      DCNSP=FS1*DCNSPP
      DCLSP=-.13*FS1
   3C CPHI=CCS(PHI)
      SPHI=SIN(PHI)
      DCZSEP=DCZSP*CPHI+DCYSP*SPHI
      DCYSEP=DCYSP*CPHI-DCZSP*SPHI
      DCMSEP=DCMSP*CPHI+DCNSP*SPHI
      DCNSEP=DCNSP*CPHI-DCMSP*SPHI
      DCLSEP=DCLSP
      GO TC 2CC
C    TABLE LCCK UP,CRUISE PHASE
C     ZERC-LIFT DRAG
  100 CDC=STCLIA(TM2,TH2,TCDC,AM,H,15,2)
C     CCNTRCL SURFACE EFFECTIVENESS
      CMULO=STCLIA(TM4,TDELT,TCMD2,AM,ADUL,5,4)
      IF (DUL.LT.0.0)  CMULC=-CMULO
      CMURC=STCLIA(TM4,TDELT,TCMD2,AM,ADUR,5,4)
      IF (DUR.GT.0.0)  CMURO=-CMURO
      CMLLC=STCLIA(TM4,TDELT,TCMD2,AM,ADLL,5,4)
      IF (DLL.LT.0.0)  CMLLC=-CMLLO
      CMLRC=STCLIA(TM4,TDELT,TCMD2,AM,ADLR,5,4)
      IF (DLR.GT.0.0)  CMLRC=-CMLRO
      CLD=CLD+.0017
C    END CF TABLE LCCK-UP
  2CC CNULC=CMLLC
      CNURC=-CMLRC
      CNLLC=-CMLLC
      CNLRC=CMLRC
      CZBC=CZBCC+DCZP
      CMBC=CMBCC+DCMP
      CABC=CDC*CCSETA
C     TRANSFORM FROM CROSS-FLCW TO BODY AXIS COMPONENTS
```

```
      CXI=COS(XIR)
      SXI=SIN(XIR)
      CAB=CABC
      CYB=CYBC*CXI-CZBC*SXI
      CZB=CYBC*SXI+CZBC*CXI
      CLB=CLBC
      CNB=CNBC*CXI-CMBC*SXI
      CMB=CNBC*SXI+CMBC*CXI
C    CONTROL SURFACE MOMENT AND FORCE COEFFICIENTS
C     CONTROL SURFACE CP IS 15.25 FT FROM NOSE
      F=(15.25-CG)/(15.25-CGR)
      CMUL=CMULC+DCMULA*ALF*DUL+DCMUL*ADUL
      CMUR=CMURC+DCMURA*ALF*DUR+DCMUR*ADUR
      CMLL=CMLLC+DCMLLA*ALP*DLL+DCMLL*ADLL
      CMLR=CMLRC+DCMLRA*ALF*DLR+DCMLR*ADLR
      CMC=(CMUL+CMUR+CMLL+CMLR)*F*PMD
      CNUL=CNULC+DCNULB*BET*DUL+DCNUL*ADUL
      CNUR=CNURC+DCNURB*BET*DUR+DCNUR*ADUR
      CNLL=CNLLC+DCNLLB*BET*DLL+DCNLL*ADLL
      CNLR=CNLRC+DCNLRB*BET*DLR+DCNLR*ADLR
      CNC=(CNUL+CNUR+CNLL+CNLR)*F*PND
      CLUL=CLD*DUL
      CLUR=CLD*DUR
      CLLL=CLD*DLL
      CLLR=CLD*DLR
      CLC=(CLUL+CLUR+CLLL+CLLR)*PLD
      CZC=-CMC*CBAR/(15.25-CG)
      CYC=-CNC*CBAR/(15.25-CG)
      CAC=CAC2*(DUL**2+DUR**2+DLL**2+DLR**2+4.*1.3*(ALP*DELQ-BET*DELR))
C    TOTAL AERODYNAMIC FORCE COEFFICIENTS
      CA=CAB+CAC
      CY=CYB+CYC+DCYSEP
      CZ=CZB+CZC+DCZSEP
      CA=-CA
      CZ=-CZ
C    TOTAL AERODYNAMIC MOMENT COEFFICIENTS
      CL=CLB+CLC+DCLSEP
      CLF=CLF*RAD*PP
      CM=CMB+CMC+CZB*(CG-CGR)/CBAR+DCMSEP
      CMC=CMC*RAD*PQ
      CN=CNB+CNC+CYB*(CG-CGR)/CBAR+DCNSEP
      CNR=CNR*RAD*PR
      RETURN
      END
C
C
C
C
```

```
      SUBROUTINE A3I
      COMMON C(3415)
      COMMON  /FLAG/  FLG1
      EQUIVALENCE (C(0601), FF1)
      EQUIVALENCE (C(0602), WEIG)
      EQUIVALENCE (C(0603), IA)
      EQUIVALENCE (C(0604), FLG2)
      FF1=0.0
      WEIG=0.0
      IA=0
      FLG1=0.0
      FLG2=0.0
      RETURN
      END
C
C
C
      SUBROUTINE A3
C
C        THIS IS THE MISSLE PROPULSION MODULE
C
C     T1=BOOST ENGINE IGNITION TIME
C     BT=BOOST ENGINE BURNOUT TIME(=T3)
C     BE=CRUISE ENGINE IGNITION TIME(=T4)
C
      COMMON  /FLAG/  FLG1
      COMMON C(3415)
      EQUIVALENCE  (C(0086) , WEIGHT )
      EQUIVALENCE (C(0110), S)
      EQUIVALENCE (C(0932),T)
      EQUIVALENCE (C(0933),T1)
      EQUIVALENCE (C(0935),BT)
      EQUIVALENCE (C(0936),BE)
      EQUIVALENCE (C(0136), CG)
      EQUIVALENCE (C(0150), OLT)
      EQUIVALENCE (C(0151), CMT)
      EQUIVALENCE (C(0152), CNT)
      EQUIVALENCE (C(0153), EPSTH)
      EQUIVALENCE (C(0154), PHITH)
      EQUIVALENCE (C(0201), A)
      EQUIVALENCE (C(0202), B)
      EQUIVALENCE (C(0203), CC)
      EQUIVALENCE (C(0073), TXBA )
      EQUIVALENCE (C(0074), TYBA )
      EQUIVALENCE (C(0075), TZBA )
      EQUIVALENCE (C(0507), H)
      EQUIVALENCE (C(0520), AMACH)
      EQUIVALENCE (C(0117), ETA)
```

```
      EQUIVALENCE (C(0601), FF1)
      EQUIVALENCE (C(0602), WEIG)
      EQUIVALENCE (C(0603), IA)
      EQUIVALENCE (C(0604), FLG2)
      EQUIVALENCE (C(0605) , DIFFM)
      EQUIVALENCE (C(0606), FLGRJ)
      EQUIVALENCE (C(0508), CD)
      EQUIVALENCE (C(2913), CEF)
      DATA WFUEL/189./
C
   80 FORMAT(1H,10X,'TOTAL ANGLE OF ATTACK EXCEEDED, ETA=',F8.3)
C
      IF (FLG1.GT.0.0) GO TO 10
      CT = 0.0
      DIFFM=0.
      FF = 0.0
      TXBA = 0.0
      TYBA = 0.0
      TZBA = 0.0
      IF (T.LT.T1) RETURN
      HIN=H*12.*.0254
      TIN=T
      AIN=ETA
      IF (T.GE.BE) GO TO 1
C   BOOST PHASE
      TT=T
      IF (T.GT.ET) GO TO 10
      CALL BOOST (TT,THRUS,FF)
      TXBA=THRUS*COS(EPSTH)
      TNBA=THRUS*SIN(EPSTH)
      TYBA=-TNBA*SIN(PHITH)
      TZBA=TNBA*COS(PHITH)
      IB=0
      WEIGHT=WEIGHT-FF+FF1
      CG=CG-(FF-FF1)/405.94
      A=A-(FF-FF1)/256.25
      B=B-(FF-FF1)/1.881
      CC=B
      CLT=0.
      CMT=TZBA*(15.525-CG)
      CNT=-TYBA*(15.525-CG)
      FF1=FF
      TW=T+DER
      IF(TW.LT.BT) GO TO 2
      IF(FLG2.GT.0.0) GO TO 2
      WEIGHT=WEIGHT-31.
      CG=CG-.15
      FLG2=1.0
```

```
      GO TO 2
   10 CONTINUE
      IF (FLGRJ.GT.0.0) GO TO 21
C     CRUISE PHASE,RAMJET OFF
      IF (IB.GE.1) GO TO 11
      CALL ENGINE(HIN,AMACH,AIN,0.,TIN,1.0,0.,0.,CT,SMARG,-1,1)
      IB=1
   11 IF (AIN.GT.9.3)  AIN=9.3
      CALL ENGINE(HIN,AMACH,AIN,0.,TIN,1.0,0.,0.,CT,SMARG,1,1)
      GO TO 4
C    SIMPLIFIED RAMJET MODEL
C    CRUISE PHASE,RAMJET OFF
   21 CALL RAMJET (HIN,AMACH,AIN,FF,1.0,CT)
      GO TO 4
 1    CONTINUE
      IF (FLGRJ.GT.0.0) GO TO 22
C     CRUISE PHASE,RAMJET ON
      IB=0
      IF (AIN.GT.10.0) GO TO 5
      FF=FF1
      IF (IA.GE.1) GO TO 3
      CALL ENGINE (HIN,AMACH,AIN,FF,TIN,0.0,0.0,0.0,CT,SMARG,-1,1)
      TT=T
      FF1=FF
      IA = 1
 3    XIN=0.0
      IF (WEIG.GT.WFUEL) XIN=1.0
      CALL ENGINE (HIN,AMACH,AIN,FF,TIN,XIN,0.0,0.0,CT,SMARG,1,1)
      GO TO 4
C    SIMPLIFIED RAMJET MODEL
C    CRUISE PHASE,RAMJET ON
   22 IF (AIN.GT.10.0) GO TO 5
      XIN=0.0
      IF (WEIG.GT.WFUEL) XIN=1.0
      CALL RAMJET (HIN,AMACH,AIN,FF,XIN,CT)
      GO TO 4
    5 IF (FLG1.GT.0.0) GO TO 10
      WRITE(6,80) ETA
      FLG1=1.0
      GO TO 10
 4    TXBA=CT*CD*S
      TYBA=0.
      TZBA=0.
      CT=T-TT
      IF (FLGRJ.GT.0.0) GO TO 6
       FF=FF/.45359237
C   DIFFUSER (INLET) MARGIN IS NOT COMPUTED IN SIMPLIFIED RAMJET MODEL
      DIFFM=SMARG/100.
```

```
    6 WEIG=WEIG+(ABS(FF+FF1))*DT/2.0
      IF (WEIG.GT.WFUEL) FLG1=1.0
      WEIGHT = WEIGHT - ABS(FF + FF1)*DT/2.
      TT=T
      FF1 = FF
      CG=CG-.00026455*ABS(FF+FF1)*DT/2.
      A=A-.001146*ABS(FF+FF1)*DT/2.
      B=B-.063492*ABS(FF+FF1)*DT/2.
      CC = B
    2 RETURN
      END
C
C
C
      SUBROUTINE A4I
C
C     ACTUATOR INITIALIZATION MODULE A4I
C
      COMMON C(3415)
      DIMENSION IPL(100)
      EQUIVALENCE (C(1735),WFIN)
      EQUIVALENCE (C(3315) ,N)
      EQUIVALENCE (C(3316), IPL(1))
      IF (WFIN.GT.0.0) GO TO 100
      GO TO 200
  100 IPL(N)=1693
      IPL(N+1)=1697
      IPL(N+2)=1701
      IPL(N+3)=1705
      IPL(N+4)=1709
      IPL(N+5)=1713
      IPL(N+6)=1717
      IPL(N+7)=1721
      N=N+8
  200 RETURN
      END
C
C
C
      SUBROUTINE A4
C
C   FIN ACTUATORS MODULE A4
C
      COMMON C(3415)
      EQUIVALENCE (C(0016),DP)
      EQUIVALENCE (C(0020),DC)
      EQUIVALENCE (C(0024),DR)
      EQUIVALENCE (C(1693),DULC)
```

```
      EQUIVALENCE (C(1696),DUL)
      EQUIVALENCE (C(1697),DURC)
      EQUIVALENCE (C(1700),DUR)
      EQUIVALENCE (C(1701),DLLC)
      EQUIVALENCE (C(1704),DLL)
      EQUIVALENCE (C(1705),DLRC)
      EQUIVALENCE (C(1708),DLR)
      EQUIVALENCE (C(1709),D1D)
      EQUIVALENCE (C(1712),D1)
      EQUIVALENCE (C(1713),D2D)
      EQUIVALENCE (C(1716),D2)
      EQUIVALENCE (C(1717),D3D)
      EQUIVALENCE (C(1720),D3)
      EQUIVALENCE (C(1721),D4D)
      EQUIVALENCE (C(1724),D4)
      EQUIVALENCE (C(1735),WFIN)
      EQUIVALENCE (C(1736),DFIN)
      EQUIVALENCE (C(1737),DCC)
      EQUIVALENCE (C(1738),DRC)
      EQUIVALENCE (C(1739),DPC)
      EQUIVALENCE (C(1740),DMAX)
      EQUIVALENCE (C(1733), DULDEG)
      EQUIVALENCE (C(1734), DULDDG)
C   ACTUATOR INPUT MIXER
      DULC=DCC+DRC+DPC
      DURC=-DCC+DRC+DPC
      DLLC=DCC-DRC+DPC
      DLRC=-DCC-DRC+DPC
C     FIN ANGLE COMMAND LIMITERS
      IF (ABS(DULC).GT.DMAX)  DULC=SIGN(DMAX,DULC)
      IF (ABS(DURC).GT.DMAX)  DURC=SIGN(DMAX,DURC)
      IF (ABS(DLLC).GT.DMAX)  DLLC=SIGN(DMAX,DLLC)
      IF (ABS(DLRC).GT.DMAX)  DLRC=SIGN(DMAX,DLRC)
C   ACTUATOR DYNAMICS
      IF (WFIN.GT.0.0) GO TO 100
      GO TO 200
C     SECOND ORDER ACTUATORS
C     RATE AND DEFLECTION LIMITS
  100 RTLIM=2.9147
C     UPPER-LEFT FIN RESPONSE
      D1D=WFIN*WFIN*(DULC-DUL)-2.*DFIN*WFIN*D1
      IF (ABS(D1).GE.RTLIM)  GO TO 110
      GO TO 120
  110 PROD1=D1D*D1
      IF (PROD1.GE.0.0)  D1D=0.0
  120 DULD=D1
C     UPPER-RT FIN RESPONSE
      D2D=WFIN*WFIN*(DURC-DUR)-2.*DFIN*WFIN*D2
```

```
      IF (ABS(D2).GE.RTLIM)  GO TO 130
      GO TO 140
  130 PRDD2=D2D*D2
      IF (PRDD2.GE.0.0)  D2D=0.0
  140 DURD=D2
C     LOWER-LEFT FIN RESPONSE
      D3D=WFIN*WFIN*(DLLC-DLL)-2.*DFIN*WFIN*D3
      IF (ABS(D3).GE.RTLIM) GO TO 150
      GO TO 160
  150 PRDD3=D3D*D3
      IF (PRDD3.GE.0.0)  D3D=0.0
  160 DLLD=D3
C     LOWER-RT FIN RESPONSE
      D4D=WFIN*WFIN*(DLRC-DLR)-2.*DFIN*WFIN*D4
      IF (ABS(D4).GE.RTLIM)  GO TO 170
      GO TO 180
  170 PROD4=D4D*D4
      IF (PRDD4.GE.0.0)  D4D=0.0
  180 DLRD=D4
      GO TO 220
C  **** IDEAL ACTUATORS ****
  200 DUL=DULC
      DUR=DURC
      DLL=DLLC
      DLR=DLRC
C  FIN ANGLE LIMITERS
      IF(ABS(DLL).GT.DMAX) DUL=SIGN(DMAX,DUL)
      IF(ABS(DLR).GT.DMAX) DLR=SIGN(DMAX,DUR)
      IF(ABS(DLL).GT.DMAX) DLL=SIGN(DMAX,DLL)
      IF(ABS(DLR).GT.DMAX) DLR=SIGN(DMAX,DLR)
C  PITCH, YAW, ROLL CHANNEL INPUTS
  220 DQ=0.25*(DUL-DUR-DLR+DLL)
      DR=0.25*(DUL+DUR-DLR-DLL)
      DP=0.25*( DUL+DUR+DLR+DLL)
      DULDEG=DLL*57.29578
      DULDEG=DLLD*57.29578
      RETURN
      END
C
C
      SUBROUTINE A5I
      RETURN
      END
C
C
      SUBROUTINE A5
      RETURN
      END
```

```
C
C
C
      SUBROUTINE C1I
C
C  INITIALIZATION MODULE FOR LVRJ AUTOPILOT
C
      COMMON C(3415)
      EQUIVALENCE (C(3315),N)
      EQUIVALENCE (C(3316),IPL(1))
      DIMENSION IPL(100)
      IPL(N)=1677
      IPL(N+1)=1689
      IPL(N+2)=1725
      IPL(N+3)=1729
      N=N+4
      RETURN
      END
C
C
C
      SUBROUTINE C1
C
C SUBROUTINE C1-LVRJ
C  THIS AUTOPILOT MODULE IS FOR STV G
C
      COMMON C(3415)
      EQUIVALENCE (C(0581), CYAP)
      EQUIVALENCE (C(0515), PSF)
      EQUIVALENCE (C(0520), AMACH)
      EQUIVALENCE (C(0415),G)
      EQUIVALENCE (C(0507),H)
      EQUIVALENCE (C(0932),T)
      EQUIVALENCE (C(0933), T1)
      EQUIVALENCE (C(0935), T3)
      EQUIVALENCE (C(1636),ANZC)
      EQUIVALENCE (C(1640),ANYC)
      EQUIVALENCE (C(1644),AZM)
      EQUIVALENCE (C(1648),AYM)
      EQUIVALENCE (C(1652),QM)
      EQUIVALENCE (C(1656),RM)
      EQUIVALENCE (C(0980), PM)
      EQUIVALENCE (C(1677),DPC1D)
      EQUIVALENCE (C(1680),DPC1)
      EQUIVALENCE (C(1689),P1D)
      EQUIVALENCE (C(1692),P1)
      EQUIVALENCE (C(1725),Z1D)
      EQUIVALENCE (C(1728),Z1)
```

```
      EQUIVALENCE (C(1729),Y1D)
      EQUIVALENCE (C(1732),Y1)
      EQUIVALENCE (C(1737),DCC)
      EQUIVALENCE (C(1738),DRC)
      EQUIVALENCE (C(1739),DPC)
      EQUIVALENCE (C(1740), DMAX)
      EQUIVALENCE (C(0165), AKNZ)
      EQUIVALENCE (C(0166), AKC)
      EQUIVALENCE (C(0167), AKNY)
      EQUIVALENCE (C(0168), AKR)
      EQUIVALENCE (C(0169), AKF)
      AKCI1=7.5
      AKRI1=7.5
      AKFI1=7.5
      AKPI2=10.
      AKCI2=3.0
      AKRI2=3.C
      QCS=-2.75/57.2957795
      QCE=1.45/57.2957795
      IF (DYNP.GT.22.5) GC TC 3
      AKNY=.1
      AKNZ=.1
      IF (DYNP.GT.4.0) GO TC 4
      AKF=.8
      AKR=1.1
      AKC=1.6
      GO TC 8
    3 AKNY=.3183502-.01573092*DYNP+3.183974E-04*DYNP**2-2.761728E-06*
     1 DYNP**3+7.692308E-09*DYNP**4
      IF (DYNP.GT.45.0) GC TC 5
      AKNZ=.4536143-.02735*DYNP+6.307143E-04*DYNP**2-5.0E-06*DYNP**3
    4 IF (DYNP.GT.35.0) GC TC 6
      AKF=1.087-.1294302*DYNP+.007295*DYNP**2-1.935556E-04*DYNP**3
     1 +1.933333E-06*DYNP**4
      AKR=1.379087-.0819641*DYNP+.00252119*DYNP**2-3.90531E-05*DYNP**3
      AKC=2.562857-.302156*DYNP+.0177683*DYNP**2-4.99414E-04*DYNP**3
     1 +5.27879E-06*DYNP**4
      GC TC 8
    5 AKNZ=.099625-.001225*DYNP
    6 AKF=.1752143-.00285476*DYNP+1.47619E-05*DYNP**2
      AKR=1.556357-.0665024*DYNP+.00112333*DYNP**2-6.66666E-06*DYNP**3
      AKC=.4083333-.0036667*DYNP
    8 IF (T.GT.T3) GO TO 1
      IF (AMACH.GT.2.0) GC TC 1
C   SEPARATION AND BOOST AUTOPILOT
      QC=QCS
      IF (T.GE.T1)  QC=QCE
      S1=1.0
```

```
      S2=0.0
      AKQI=AKQI1
      AKRI=AKRI1
      AKPI=AKPI1
      GO TO 16
C   CRUISE ALTCFILCT
    1 CONTINUE
      S1=0.0
      S2=1.0
      AKQI=AKQI2
      AKRI=AKRI2
      AKPI=AKPI2
   16 CONTINUE
C     PITCH AND YAW CHANNELS
      Q1=AKQ*(QM-S1*QC)
      R1=AKR*RM
      ANZM=-AZM/G
      ANYM=AYM/G
      ANZ=ANZC-ANZM
      ANY=ANYC-ANYM
C      POS DQC AND DRC COMMAND NEG TURNING MOMENTS
      DQC1=Q1-S2*AKNZ*ANZ
      Z1D=DQC1
      IF (ABS(DQC).GE.DMAX)  Z1D=0.0
      DQC=DQC1+AKQI*Z1
      IF (ABS(DQC).GE.DMAX)  DQC=SIGN(DMAX,DQC)
      DRC1=R1-S2*AKNY*ANY
      Y1D=DRC1
      IF (ABS(DRC).GE.DMAX)  Y1D=0.0
      DRC=DRC1+AKRI*Y1
      IF (ABS(DRC).GE.DMAX)  DRC=SIGN(DMAX,DRC)
    2 CONTINUE
C      ROLL CHANNEL
      P1D=AKPI*PM
C      LAG COMPENSATOR
      OMG1=.1
      OMG2=.01
      DPCD=-AKP*(PM+P1)*OMG2/OMG1
      DPCID=(OMG1-OMG2)*DPCD-OMG2*DPCI
      DPC=DPCI+DPCD
      IF (ABS(DPC).GE.DMAX)  DPC=SIGN(DMAX,DPC)
      RETURN
      END
C
C
C
      SUBROUTINE C2I
C
```

```
      DIMENSION IPL(100)
      COMMON C(3415)
      EQUIVALENCE (C(0507),H)
      EQUIVALENCE (C(3315),N)
      EQUIVALENCE (C(3316),IPL(1))
      IPL(N)=1633
      IPL(N+1)=1637
      N=N+2
      HNCW=H
      RETURN
      END
C
C
C
      SUBROUTINE C2
C
C   GUIDANCE COMMAND MODULE
C
      COMMON C(3415)
      EQUIVALENCE (C(0256), A33)
      EQUIVALENCE (C(0286), VXTP)
      EQUIVALENCE (C(0294), VYTP)
      EQUIVALENCE (C(0302), VZTP)
      EQUIVALENCE (C(337), CMGYC)
      EQUIVALENCE (C(338), CMGZC)
      EQUIVALENCE (C(0317), FLGT)
      EQUIVALENCE (C(0339), XMA)
      EQUIVALENCE (C(0340), YMA)
      EQUIVALENCE (C(0341), ZMA)
      EQUIVALENCE (C(0520), AMACH)
      EQUIVALENCE (C(0500),HCC)
      EQUIVALENCE (C(0501),HREF)
      EQUIVALENCE (C(0502),HD)
      EQUIVALENCE (C(0507),H)
      EQUIVALENCE (C(0508), CD)
      EQUIVALENCE (C(0515),PSF)
      EQUIVALENCE (C(0527),GAMMAV)
      EQUIVALENCE (C(0528), VT)
      EQUIVALENCE (C(0531),VH)
      EQUIVALENCE (C(0581), CYNP)
      EQUIVALENCE (C(0582),AKG)
      EQUIVALENCE (C(0932),T)
      EQUIVALENCE (C(0933),T1)
      EQUIVALENCE (C(0935),T3)
      EQUIVALENCE (C(0937),T5)
      EQUIVALENCE (C(942),TCC)
      EQUIVALENCE (C(0944),TLD)
      EQUIVALENCE (C(0945),TFC)
```

```
      EQUIVALENCE (C(1633),ANZCD)
      EQUIVALENCE (C(1636),ANZC)
      EQUIVALENCE (C(1637),ANYCD)
      EQUIVALENCE (C(1640),ANYC)
      EQUIVALENCE (C(1750), WF1)
      DATA RCON/36500./
      DYNP=QC/144.
C   ZERO MODE CONTROL SWITCHES
      S4=0.0
      S5=0.0
      S6=0.0
      S8=0.0
      IF (T.GE.T3) GO TO 1
C   SEPARATION,BOOST PHASE
      IF(AMACH.GE.2.0) GO TO 1
      GO TO 100
C   ZERO G TRAJECTORY
    1 IF(T.GE.T5) GO TO 2
      GO TO 100
C   LEVEL FLIGHT
    2 IF(T.GE.TDC) GO TO 5
      S4=1.0
      S5=1.0
      GO TO 100
C   DIVE,CLIMB
    5 IF(T.GE.TLD) GO TO 4
      HDC1=HDC
    7 S4=1.0
      S5=1.0
      S8=1.0
      GO TO 100
C   AUTOMATIC LETDOWN TO HREF
    4 HORIZ=SQRT(XMA**2+YMA**2)
      D1=1.575*VH+RCON-HORIZ
      IF (D1.GE.0.0) GO TO 6
      S4=1.0
      S5=1.0
      S6=1.0
      GO TO 100
C     TERMINAL DIVE
    6 W3=(ZMA*VXTP-XMA*VZTP)/(XMA*VXTP+ZMA*VZTP)
      ANZC1=-2.*2900.*W3/32.172+A33
      IF (FLGT.GT.0.0) GO TO 8
      GO TO 108
C     TERMINAL HOMING
    8 ANZC1=ANG*VT*OMGYC/32.174
      GO TO 108
  100 CONTINUE
```

```
C   ***** VERTICAL CHANNEL *****
C   HERR LIMITER
        HERR1=S6*(H-HREF)
        HERR2=0.5*HERR1
        IF (ABS(HERR1).GT.800.)  HERR2=SIGN(400.,HERR1)
        ANZC1=.0333*(S8*HDC1-S4*(HD+HERR2))+S5*A33
C   NZ LIMITER
        IF(ANZC1) 108,107,108
  108 AZMAX=.35*(DYNP-5.714)
        IF (DYNP.LE.30.0) AZMAX=8.5*DYNP/30.
        AZMAX=AMIN1(AZMAX,12.0)
        IF (ABS(ANZC1).GE.AZMAX)  ANZC1=SIGN(AZMAX,ANZC1)
  107 CONTINUE
C   ***** LATERAL CHANNEL *****
        ANYC1=0.
        IF (T.LT.T5) GO TO 210
        IF (FLGT.GT.0.0) GO TO 200
C     CROSS-PRODUCT MIDCOURSE GUIDANCE
        CPROD=(YMA*VXTP-XMA*VYTP)/(XMA*VXTP+YMA*VYTP)
        ANYC1=2900.*CPROD/32.172
        GO TO 205
C       TERMINAL HOMING
  200 ANYC1=AKG*VT*OMGZC/32.172
C     NY LIMITER
  205 IF (ANYC1) 208,207,208
  208 AYMAX=8.0*(DYNP-11.25)/22.5
        IF (DYNP.LE.22.5) AYMAX=4.0*DYNP/22.5
        AYMAX=AMIN1(AYMAX,12.0)
        IF (ABS(ANYC1).GT.AYMAX)  ANYC1=SIGN(AYMAX,ANYC1)
  207 CONTINUE
C     ELLIPTICAL LIMITING
        DENOM=SQRT((AZMAX*ANYC1)**2+(AYMAX*ANZC1)**2)
        IF (DENOM.LE.0.0)  GO TO 210
        AZLIM=AZMAX*AYMAX*ANZC1/DENOM
        IF (ABS(ANZC1).GE.ABS(AZLIM))  ANZC1=AZLIM
        AYLIM=AZMAX*AYMAX*ANYC1/DENOM
        IF (ABS(ANYC1).GE.ABS(AYLIM))  ANYC1=AYLIM
C LATERAL COMMAND FILTER
  210 ANYCD=WF1*(ANYC1-ANYC)
C     VERTICAL COMMAND FILTER
        ANZCD=WF1*(ANZC1-ANZC)
        RETURN
        END
C
C
        SUBROUTINE C3I
        RETURN
        END
```

```
C
C
      SUBROUTINE C3
      RETURN
      END
C
C
      SUBROUTINE C4I
      RETURN
      END
C
C
      SUBROUTINE C4
      RETURN
      END
C
C
      SUBROUTINE C5I
      RETURN
      END
C
C
      SUBROUTINE C5
      RETURN
      END
C
C
      SUBROUTINE C6I
      RETURN
      END
C
C
      SUBROUTINE C6
      RETURN
      END
C
C
      SUBROUTINE C7I
      RETURN
      END
C
C
      SUBROUTINE C7
      RETURN
      END
C
C
      SUBROUTINE C8I
```

```
      RETURN
      END
C
C
      SUBROUTINE C8
      RETURN
      END
C
C
      SUBROUTINE C9I
      RETURN
      END
C
C
      SUBROUTINE C9
      RETURN
      END
C
C
      SUBROUTINE C10I
      RETURN
      END
C
C
      SUBROUTINE C10
      RETURN
      END
C
C
C
      SUBROUTINE D1I                                             D1IA0030
C
C     TRANSLATIONAL DYNAMICS INITIALIZATION MODULE D1AB          D1IA0010
C     FOR USE WITH MODULE D1A OR D1B                             D1IA0020
C
      DIMENSION  IPL (100)                                       D1IA0070
      COMMON C(3415)                                             D1IA0040
      EQUIVALENCE  (C(3315) , N     )                            D1IA0050
      EQUIVALENCE (C(3316),IPL(1))
      IPL(N) = 283                                               D1IA0100
      IPL(N+1) = 291                                             D1IA0110
      IPL(N+2) = 299                                             D1IA0120
      IPL(N+3) = 287                                             D1IA0130
      IPL(N+4) = 295                                             D1IA0140
      IPL(N+5) = 303                                             D1IA0150
      N = N+6                                                    D1IA0160
      RETURN                                                     D1IA0170
      END                                                        D1IA0200
```

```
C
C
C     SUBROUTINE D1
C
C     TRANSLATIONAL DYNAMICS MODULE D1B - BODY AXES
C
      COMMON C(3415)
      EQUIVALENCE  (C(0073) , TXBA    )
      EQUIVALENCE  (C(0074) , TYBA    )
      EQUIVALENCE  (C(0075) , TZBA    )
      EQUIVALENCE  (C(0086) , WE      )
      EQUIVALENCE (C(0112), FEJ)
      EQUIVALENCE  (C(0128) , FXBA    )
      EQUIVALENCE  (C(0129) , FYBA    )
      EQUIVALENCE  (C(0130) , FZBA    )
      EQUIVALENCE (C(0224),A11   )
      EQUIVALENCE (C(0228),A12   )
      EQUIVALENCE (C(0232),A13   )
      EQUIVALENCE (C(0236),A21   )
      EQUIVALENCE (C(0240),A22   )
      EQUIVALENCE (C(0244),A23   )
      EQUIVALENCE (C(0248),A31   )
      EQUIVALENCE (C(0252),A32   )
      EQUIVALENCE (C(0256),A33   )
      EQUIVALENCE  (C(0282) , AG      )
      EQUIVALENCE  (C(0283) , VXDTP   )
      EQUIVALENCE  (C(0286) , VXTP    )
      EQUIVALENCE  (C(0287) , XDTP    )
      EQUIVALENCE  (C(0290) , XTP     )
      EQUIVALENCE  (C(0291) , VYDTP   )
      EQUIVALENCE  (C(0294) , VYTP    )
      EQUIVALENCE  (C(0295) , YDTP    )
      EQUIVALENCE  (C(0298) , YTP     )
      EQUIVALENCE  (C(0299) , VZDTP   )
      EQUIVALENCE  (C(0302) , VZTP    )
      EQUIVALENCE  (C(0303) , ZDTP    )
      EQUIVALENCE  (C(0306) , ZTP     )
      EQUIVALENCE  (C(0411) , GCX     )
      EQUIVALENCE  (C(0412) , GCY     )
      EQUIVALENCE  (C(0413) , GCZ     )
      EQUIVALENCE  (C(0710) , TAXBA   )
      EQUIVALENCE  (C(0711) , TAYBA   )
      EQUIVALENCE  (C(0712) , TAZBA   )
      EQUIVALENCE  (C(0713) , TAX)
      EQUIVALENCE (C(0714)  , TAY)
      EQUIVALENCE  (C(0715)  , TAZ)
      AGWE = AG/WE
```

```
C     TOTAL ACCELERATION DUE TO AERODYNAMIC FORCES AND THRUST FORCES
      TAXBA = AGWE*(FXBA + TXBA)
      TAYBA = AGWE*(FYBA + TYBA)
      TAZBA= AGWE*(FZBA+TZBA+FEJ)
C     RESOLVE FROM BODY AXES TO TANGENT PLANE
      TAX = A11*TAXBA + A21*TAYBA + A31*TAZBA
      TAY = A12*TAXBA + A22*TAYBA + A32*TAZBA
      TAZ = A13*TAXBA + A23*TAYBA + A33*TAZBA
C     INTEGRATE ACCELERATIONS DUE TO AERODYNAMICS, THRUST, GRAVITY,
C     AND CORIOLIS
      VXDTP = TAX + GCX
      VYDTP = TAY + GCY
      VZDTP = TAZ + GCZ
C     INTEGRATE VELOCITY
      XDTP = VXTP
      YDTP = VYTP
      ZDTP = VZTP
      RETURN
      END
C
C
C
      SUBROUTINE D2I
C
C     ROTATIONAL DYNAMICS INITIALIZATION MODULE D2IEUL
C     FOR USE WITH MODULE D2F OR D2SA
C
      COMMON C(3415)
      EQUIVALENCE (C(224) ,A11    )
      EQUIVALENCE (C(228) ,A12    )
      EQUIVALENCE (C(232) ,A13    )
      EQUIVALENCE (C(236) ,A21    )
      EQUIVALENCE (C(240) ,A22    )
      EQUIVALENCE (C(244) ,A23    )
      EQUIVALENCE (C(248) ,A31    )
      EQUIVALENCE (C(252) ,A32    )
      EQUIVALENCE (C(256) ,A33    )
      EQUIVALENCE (C(2901),ROLLO )
      EQUIVALENCE (C(2902),PITCHO)
      EQUIVALENCE (C(2903),YAWO  )
      EQUIVALENCE  (C(3315) , N     )
      DIMENSION  IPL (100)
      SP=SIN(ROLLO)
      CT=COS(PITCHO)
      ST=SIN(PITCHO)
      EQUIVALENCE  (C(3316) , IPL(1))
      SS = SIN(YAWO)
      CS=COS(YAWO)
```

```
      CP = COS(ROLLO)
      A11= CS*CT
      A12= SS*CT
      A13= -ST
      A21= -SS*CP + CS*ST*SP
      A22= CS*CP +SS*ST*SP
      A23= CT*SP
      A31= CS*ST*CP + SS*SP
      A32= SS*ST*CP - CS*SP
      A33= CT*CP
      IPL(N) = 205
      IPL(N+1) = 213
      IPL(N+2) = 217
      IPL(N+3) = 221
      IPL(N+4) = 225
      IPL(N+5) = 229
      IPL(N+6) = 233
      IPL(N+7) = 237
      IPL(N+8) = 241
      IPL(N+9) = 245
      IPL(N+10) = 249
      IPL(N+11) = 253
      N = N+12
      RETURN
      END
C
C
C
      SUBROUTINE D2
C
C     ROTATIONAL DYNAMICS MODULE D2PA- PRINCIPAL AXES
C
      COMMON C(3415)
      COMMON/AFFLE/TM1(7),TM2(15),TM3(14),TM4(5),TETA1(7),TETA2(7),
     1TETA3(6),TXI1(2),TXI2(10),TXI3(5),TDELT(4),TH1(3),TH2(2),
     2TCZBC(7,2,7),TDCZP(14,7),TCYBC(7,10,6),TCMBC(7,5,7),TDCMF(14,7),
     3TCDB(15,2),TCDC(15,2),TCAD2(15),TCNBC(7,10,6),TCMD1(15,4),
     4TCMD2(5,4),TCCMD(15),TDCMDA(15),TPMD(14,3),TCMQ(15),TPQ(14,3),
     5TCLBC(7,10,6),TCLD(15),TFLD(14,3),TCLP(15),TFP(14,3),TDCYSP(3),
     LTDCZSP(3),TCCMSP(3),TCCNSP(3),TH3(3),TFEJ(13),TTEJ(13)
      EQUIVALENCE (C(0109), FLGSEP)
      EQUIVALENCE (C(0112), FEJ)
      EQUIVALENCE  (C(0131) , CLBA    )
      EQUIVALENCE  (C(0132) , CMBA    )
      EQUIVALENCE  (C(0133) , CNBA    )
      EQUIVALENCE (C(0136), CG)
      EQUIVALENCE  (C(0150) , CLT     )
      EQUIVALENCE  (C(0151) , CMT     )
```

```
      EQUIVALENCE   (C(0152) , CNT     )
      EQUIVALENCE   (C(0201) , A       )
      EQUIVALENCE   (C(0202) , B       )
      EQUIVALENCE   (C(0209) , PDBA    )
      EQUIVALENCE   (C(0203) , G       )
      EQUIVALENCE   (C(0212) , PBA     )
      EQUIVALENCE   (C(0213) , QDBA    )
      EQUIVALENCE   (C(0216) , QBA     )
      EQUIVALENCE   (C(0217) , RDBA    )
      EQUIVALENCE   (C(0220) , RBA     )
      EQUIVALENCE   (C(0221) , AD11    )
      EQUIVALENCE   (C(0224) , A11     )
      EQUIVALENCE   (C(0228) , A12     )
      EQUIVALENCE   (C(0225) , AD12    )
      EQUIVALENCE   (C(0229) , AD13    )
      EQUIVALENCE   (C(0232) , A13     )
      EQUIVALENCE   (C(0236) , A21     )
      EQUIVALENCE   (C(0237) , AD22    )
      EQUIVALENCE   (C(0233) , AD21    )
      EQUIVALENCE   (C(0240) , A22     )
      EQUIVALENCE   (C(0241) , AD23    )
      EQUIVALENCE   (C(0245) , AD31    )
      EQUIVALENCE   (C(0244) , A23     )
      EQUIVALENCE   (C(0248) , A31     )
      EQUIVALENCE   (C(0249) , AD32    )
      EQUIVALENCE   (C(0252) , A32     )
      EQUIVALENCE (C(0253), AD33)
      EQUIVALENCE (C(0256), A33 )
      EQUIVALENCE (C(0932), T )
      EQUIVALENCE (C(0933), T1)
      FEJ=0.0
      XEJ=10.585
      IF (T1.LE.0.0)  FLGSEP=0.0
      YEJ=-.01
      IF (FLGSEP.LE.0.) GO TO 10
      IF (T.GE.0.1) GO TO 10
C       EJECTION FORCE
      FEJ=CDLI(T,TTEJ,TFEJ,13)
C     ADD AERODYNAMIC,THRUST,AND EJECTION MOMENTS
   10 TLBA=CLBA+CLT+FEJ*YEJ
      TMBA=CMBA+CMT+FEJ*(XEJ-CG)
      TNBA = CNBA + CNT
C     INTEGRATE BODY ANGULAR RATES
      PDBA = (TLBA + (B-G)*QBA*RBA)/A
      QDBA = (TMBA + (G-A)*RBA*PBA)/B
      RDBA = (TNBA + (A-B)*PBA*QBA)/G
C     INTEGRATE ATTITUDE DIRECTION COSINES
      AD11=A21*RBA-A31*QBA
```

```
      AD22=A32*PBA-A12*RBA
      AD21=A31*FBA-A11*RBA
      AD13=A23*RBA-A33*OBA
      AD12=A22*FBA-A32*CBA
      AD33=A13*CBA-A23*PBA
      AD32=A12*CBA-A22*PBA
      AD31=A11*CBA-A21*PBA
      AD23=A33*PBA-A13*RBA
      RETURN
      END
C
C
      SUERCUTINE D3I
      RETURN
      END
C
C
      SUBRCUTINE D3
      RETURN
      END
C
C
      SUBRCUTINE D4I
      RETURN
      END
C
C
      SUBRCUTINE D4
      RETURN
      END
C
C
      SUBRCUTINE D5I
      RETURN
      ENC
C
C
      SUBROUTINE D5
      RETURN
      ENC
C
C
C
      SUBRCUTINE G1I
C
C     GRAVITATICNAL AND CCRICLIS ACCELERATION INITIALIZATION MODULE G1IC
C
      CCMMCN C(34151)
```

```
      EQUIVALENCE  (C(0402) , ZS          )
      EQUIVALENCE  (C(0403) , GM          )
      EQUIVALENCE  (C(0405) , ER          )
      EQUIVALENCE  (C(0406) , ALPO        )
      EQUIVALENCE  (C(0407) , CLAMO       )
      EQUIVALENCE  (C(0408) , CMEGX       )
      EQUIVALENCE  (C(0409) , CMEGY       )
      EQUIVALENCE  (C(0410) , CMEGZ       )
      EQUIVALENCE  (C(0414) , FO          )
      EQUIVALENCE  (C(0415) , GO          )
      EQUIVALENCE  (C(0503) , RE          )
      ZS = RE + HO
      GM = GO*ZS*ZS
      RT = ER + ER
      CMEGZ = -RT*SIN(CLAMO)
      RT = RT*COS(CLAMO)
      CMEGY = -RT*SIN(ALPO)
      CMEGX = RT*COS(ALPO)
      RETURN
      END
C
C
C
      SUBROUTINE G1
C
C     GRAVITATIONAL AND CORIOLIS ACCELERATION MODULE GIC
C
      COMMON C(3415)
      EQUIVALENCE  (C(0286) , VXTP   )
      EQUIVALENCE  (C(0294) , VYTP   )
      EQUIVALENCE  (C(0302) , VZTP   )
      EQUIVALENCE  (C(0290) , XTP         )
      EQUIVALENCE  (C(0298) , YTP         )
      EQUIVALENCE  (C(0306) , ZTP         )
      EQUIVALENCE  (C(0401) , R           )
      EQUIVALENCE  (C(0402) , ZS          )
      EQUIVALENCE  (C(0403), GM)
      EQUIVALENCE  (C(0404) , GZRO        )
      EQUIVALENCE  (C(0408) , CMEGX       )
      EQUIVALENCE  (C(0409) , CMEGY       )
      EQUIVALENCE  (C(0410) , CMEGZ       )
      EQUIVALENCE  (C(0411) , GCX         )
      EQUIVALENCE  (C(0412) , GCY         )
      EQUIVALENCE  (C(0413) , GCZ         )
      EQUIVALENCE  (C(0414) , HO          )
      EQUIVALENCE  (C(0415) , GO          )
      EQUIVALENCE (C(0502),HD)
      EQUIVALENCE  (C(0503) , RE          )
```

```
      EQUIVALENCE  (C(0507) , H         )
      EQUIVALENCE  (C(0507), H )
C     IF GZRC = 0, FLAT-EARTH GRAVITATIONAL FIELD
      IF (GZRC .EQ. 1.0) GO TO 1
C     COMPUTE FLAT-EARTH GRAVITATIONAL ACCELERATION
      GCX = 0.
      GCY = 0.
      GCZ = GC
      H = -ZTP +HC
      HD=-VZTP
      RETURN
C     IF GZRC = 1, SPHERICAL GRAVITATIONAL FIELD
    1 R = SQRT(XTP*XTP + YTP*YTP + (ZTP-ZS)*(ZTP-ZS))
      HD=(XTP*VXTP+YTP*VYTP+(ZTP-ZS)*VZTP)/R
      H = R - RE
C     COMPUTE SPHERICAL GRAVITATIONAL ACCELERATION
      GMR = -GM/(R*R*R)
      GX = GMR*XTP
      GY = GMR*YTP
      GZ = GMR*(ZTP-ZS)
C     ADD CORIOLIS ACCELERATION
      GCX = GX + OMEGZ*VYTP - OMEGY*VZTP
      GCY = GY + OMEGX*VZTP - OMEGZ*VXTP
      GCZ = GZ + OMEGY*VXTP - OMEGX*VYTP
      RETURN
      END
C
C
      SUBROUTINE G2I
      RETURN
      END
C
C
      SUBROUTINE G2
      RETURN
      END
C
C
C
      SUBROUTINE G3I
C
      COMMON C(3415)
      EQUIVALENCE (C(0208) , TSA )
      EQUIVALENCE (C(0257) , CTS )
      EQUIVALENCE (C(0258) , STS )
      EQUIVALENCE (C(0306), ZTP)
      EQUIVALENCE (C(0414), HC)
      EQUIVALENCE (C(0507), H)
```

```
      EQUIVALENCE (C(0515), FSF)
      EQUIVALENCE (C(0529), VAT)
      EQUIVALENCE (C(0551), RHCH)
      EQUIVALENCE (C(0561), VAH)
      H=HC-ZTF
      ALT=H/3.280839895
      VEL=VAT/3.280839895
      CALL AIR (ALT,VEL,-1,P,T,SON,DEN,Q,G,1,2)
      CTS = COS(TSA)
      STS = SIN(TSA)
      RETURN
      END
C
C
C
      SUBROUTINE G3
C
C     AIR DATA MODULE G3
C
      COMMON C(3415)
      EQUIVALENCE  (C(0224) , A11        )
      EQUIVALENCE  (C(0228) , A12        )
      EQUIVALENCE  (C(0232) , A13        )
      EQUIVALENCE  (C(0236) , A21        )
      EQUIVALENCE  (C(0240) , A22        )
      EQUIVALENCE  (C(0244) , A23        )
      EQUIVALENCE  (C(0248) , A31        )
      EQUIVALENCE  (C(0252) , A32        )
      EQUIVALENCE  (C(0256) , A33        )
      EQUIVALENCE  (C(0257) , CTS        )
      EQUIVALENCE  (C(0258) , STS        )
      EQUIVALENCE  (C(0282) , AG         )
      EQUIVALENCE  (C(0286) , VXTP       )
      EQUIVALENCE  (C(0294) , VYTP       )
      EQUIVALENCE  (C(0302) , VZTP       )
      EQUIVALENCE  (C(0451) , TWXTP      )
      EQUIVALENCE  (C(0452) , TWYTP      )
      EQUIVALENCE  (C(0453) , TWZTP      )
      EQUIVALENCE  (C(0504) , VATX       )
      EQUIVALENCE  (C(0505) , VATY       )
      EQUIVALENCE  (C(0506) , VATZ       )
      EQUIVALENCE  (C(0507) , H          )
      EQUIVALENCE  (C(0508) , CD         )
      EQUIVALENCE  (C(0509) , UBA        )
      EQUIVALENCE  (C(0510) , VBA        )
      EQUIVALENCE  (C(0511) , WBA        )
      EQUIVALENCE  (C(0512) , USA        )
      EQUIVALENCE  (C(0513) , VSA        )
```

```
      EQUIVALENCE  (C(0514) , WSA        )
      EQUIVALENCE (C(0515), PSF)
      EQUIVALENCE  (C(0518) , ASA        )
      EQUIVALENCE  (C(0519) , ESA        )
      EQUIVALENCE  (C(0520) , AMACH      )
      EQUIVALENCE  (C(0529) , VAT        )
      EQUIVALENCE (C(0551) , RHCH)
      EQUIVALENCE (C(0561) , VAH)
C     VELOCITY WITH RESPECT TO AIR MASS
      VATX = VXTP - TWXTP
      VATY = VYTP - TWYTP
      VATZ = VZTP - TWZTP
      VATSQ = VATX*VATX + VATY*VATY + VATZ*VATZ
      VAT = SQRT(VATSQ)
      ALT=H/3.280839895
      VEL=VAT/3.280839895
      CALL AIR (ALT,VEL,1,P,T,SON,DEN,Q,G,1,2)
      PSF=.020885434*P
      RHCH=.062427961*DEN
      CD=.020885434*Q
      VAH=3.280839895*SON
      AMACH = VAT/VAH
C    ORTHOGONALIZE DIRECTION COSINES
      DAN=SQRT(A11**2+A12**2+A13**2)
      E11=A11/DAN
      E12=A12/DAN
      E13=A13/DAN
      D=E11*A21+E12*A22+E13*A23
      V21=A21-D*E11
      V22=A22-D*E12
      V23=A23-D*E13
      DON=SQRT(V21**2+V22**2+V23**2)
      E21=V21/DON
      E22=V22/DON
      E23=V23/DON
      C1=E11*A31+E12*A32+E13*A33
      C2=E21*A31+E22*A32+E23*A33
      V31=A31-C1*E11-C2*E21
      V32=A32-C1*E12-C2*E22
      V33=A33-C1*E13-C2*E23
      DAP=SQRT(V31**2+V32**2+V33**2)
      E31=V31/DAP
      E32=V32/DAP
      E33=V33/DAP
      A11=E11
      A12=E12
      A13=E13
      A21=E21
```

```
      A22=E22
      A23=E23
      A31=E31
      A32=E32
      A33=E33
C     VELOCITY IN BODY AXES
      UBA=A11*VATX+A12*VATY+A13*VATZ
      VBA=A21*VATX+A22*VATY+A23*VATZ
      WBA = A31*VATX + A32*VATY + A33*VATZ
C     VELOCITY IN STABILITY AXES
      USA=CTS*UBA+STS*WBA                                          T)Y0126C
      VSA=VBA                                                      T)Y0127C
      WSA=-STS*UBA+CTS*WBA                                         T)Y0128C
C     ANGLE OF ATTACK AND SIDESLIP
      ASA = ATAN(WSA/USA)
      BSA = ARSIN(VSA/SQRT(USA*USA + VSA*VSA + WSA*WSA))
      RETURN
      END
C
C
      SUBROUTINE G4I
      RETURN
      END
C
C
      SUBROUTINE G4
      RETURN
      END
C
C
C
      SUBROUTINE G5I
C
C   COORDINATE CONVERSION INITIALIZATION MODULE
C
      COMMON C(3415)
      EQUIVALENCE  (C(0557),CTPI11)
      EQUIVALENCE  (C(0558),CTPI12)
      EQUIVALENCE  (C(0559),CTPI13)
      EQUIVALENCE  (C(0560),CTPI21)
      EQUIVALENCE  (C(0562),CTPI22)
      EQUIVALENCE  (C(0563),CTPI23)
      EQUIVALENCE  (C(0564),CTPI31)
      EQUIVALENCE  (C(0565),CTPI32)
      EQUIVALENCE  (C(0566),CTPI33)
      EQUIVALENCE  (C(0576),ALATI)
      EQUIVALENCE  (C(0578),AZ)
      AZ = AZ*.01745329
```

```
      ALAT = ALAT*.01745329
      CTPI11 =-COS(AZ)*SIN(ALAT)
      CTPI12 =   SIN(ALAT)*SIN(AZ)
      CTPI13 = -COS(ALAT)
      CTPI21 = SIN(AZ)
      CTPI22 =   COS(AZ)
      CTPI23 = 0.0
      CTPI31 =   COS(ALAT)*COS(AZ)
      CTPI32 =-COS(ALAT)*SIN(AZ)
      CTPI33 = -SIN(ALAT)
      RETURN
      END
C
C
      SUBROUTINE G5
C
C     COORDINATE CONVERSION MODULE G5
C
      COMMON C(3415)
      EQUIVALENCE  (C(0224),CTPB11)
      EQUIVALENCE  (C(0228),CTPB12)
      EQUIVALENCE  (C(0232),CTPB13)
      EQUIVALENCE  (C(0236),CTPB21)
      EQUIVALENCE  (C(0240),CTPB22)
      EQUIVALENCE  (C(0244),CTPB23)
      EQUIVALENCE  (C(0248),CTPB31)
      EQUIVALENCE  (C(0252),CTPB32)
      EQUIVALENCE  (C(0256),CTPB33)
      EQUIVALENCE  (C(0557),CTPI11)
      EQUIVALENCE  (C(0558),CTPI12)
      EQUIVALENCE  (C(0559),CTPI13)
      EQUIVALENCE  (C(0560),CTPI21)
      EQUIVALENCE  (C(0562),CTPI22)
      EQUIVALENCE  (C(0563),CTPI23)
      EQUIVALENCE  (C(0564),CTPI31)
      EQUIVALENCE  (C(0565),CTPI32)
      EQUIVALENCE  (C(0566),CTPI33)
      EQUIVALENCE  (C(0567),CBI11)
      EQUIVALENCE  (C(0568),CBI12)
      EQUIVALENCE  (C(0569),CBI13)
      EQUIVALENCE  (C(0569),CBI13)
      EQUIVALENCE  (C(0570),CBI21)
      EQUIVALENCE  (C(0571),CBI22)
      EQUIVALENCE  (C(0572),CBI23)
      EQUIVALENCE  (C(0573),CBI31)
      EQUIVALENCE  (C(0574),CBI32)
      EQUIVALENCE  (C(0575),CBI33)
      EQUIVALENCE  (C(0710),TAXAA)
```

```
      EQUIVALENCE  (C(0711),TAYAA)
      EQUIVALENCE  (C(0712),TAZAA)
      EQUIVALENCE  (C(0286), VXTP)
      EQUIVALENCE  (C(0294), VYTP)
      EQUIVALENCE  (C(0302), VZTP)
      EQUIVALENCE  (C(0290) , XTP          )
      EQUIVALENCE  (C(0298) , YTP          )
      EQUIVALENCE  (C(0306) , ZTP          )
      EQUIVALENCE  (C(0521) , THET         )
      EQUIVALENCE  (C(0554),AXECI)
      EQUIVALENCE  (C(0555),AYECI)
      EQUIVALENCE  (C(0556),AZECI)
      EQUIVALENCE  (C(0577),VXECI)
      EQUIVALENCE  (C(0552),VYECI)
      EQUIVALENCE  (C(0553),VZECI)
      EQUIVALENCE  (C(0547),XECI)
      EQUIVALENCE  (C(0548),YECI)
      EQUIVALENCE  (C(0549),ZECI)
      EQUIVALENCE  (C(0402),ZS)
      EQUIVALENCE  (C(0522) , PSI          )
      EQUIVALENCE  (C(0523) , PHI          )
      EQUIVALENCE  (C(0524) , BEAR         )
      EQUIVALENCE  (C(0525) , ELEV         )
      EQUIVALENCE  (C(0526) , SLANT        )
      EQUIVALENCE  (C(0527), GAMMAV)
      EQUIVALENCE  (C(0528), VGS)
      EQUIVALENCE  (C(0531), VH)
      EQUIVALENCE  (C(0532), GAMMAH)
C MATRIX A = MATRIX CTPB
C     HEADING, PITCH, ROLL
      PSI = ATAN2(CTPB12,CTPB11)
      THET = -ARSIN(CTPB13)
      PHI = ATAN2(CTPB23,CTPB33)
C     FLIGHT PATH
      VH= SQRT(VXTP*VXTP + VYTP*VYTP)
      GAMMAH = ATAN2(VYTP,VXTP)
      GAMMAV = ATAN (-VZTP/VH)
C     SLANT RANGE AND TOTAL VELOCITY
      SLANT = SQRT(XTP*XTP + YTP*YTP + ZTP*ZTP)
      VGS = SQRT(VXTP*VXTP + VYTP*VYTP + VZTP*VZTP)
C     BEARING AND ELEVATION
      IF(YTP) 3,1,3
    1 IF(XTP) 3,2,3
    2 BEAR=0.0
      GO TO 4
    3 BEAR=ATAN2(YTP,XTP)
    4 CONTINUE
      IF(XTP.EQ.0.0 .AND. YTP.EQ.0.0) GO TO 50
```

```
      ELEV = ATAN (-ZTP/SQRT(XTP*XTP + YTP*YTP))
      GO TO 60
   50 ELEV=0.0
   60 CONTINUE
C   CEI = CTPI*CETP
      CBI11 = CTPI11*CTPB11 + CTPI12*CTPB12 + CTPI13*CTPB13
      CBI12 = CTPI11*CTPB21 + CTPI12*CTPB22 + CTPI13*CTPB23
      CBI13 = CTPI11*CTPB31 + CTPI12*CTPB32 + CTPI13*CTPB33
      CBI21 = CTPI21*CTPB11 + CTPI22*CTPB12 + CTPI23*CTPB13
      CBI22 = CTPI21*CTPB21 + CTPI22*CTPB22 + CTPI23*CTPB23
      CBI23 = CTPI21*CTPB31 + CTPI22*CTPB32 + CTPI23*CTPB33
      CBI31 = CTPI31*CTPB11 + CTPI32*CTPB12 + CTPI33*CTPB13
      CBI32 = CTPI31*CTPB21 + CTPI32*CTPB22 + CTPI33*CTPB23
      CBI33 = CTPI31*CTPB31 + CTPI32*CTPB32 + CTPI33*CTPB33
C   DETERMINE ACCELERATION, VELOCITY AND POSITION IN ECI SYSTEM
      AXECI = CTPI11*TAXAA + CTPI12*TAYAA + CTPI13*TAZAA
      AYECI = CTPI21*TAXAA + CTPI22*TAYAA + CTPI23*TAZAA
      AZECI = CTPI31*TAXAA + CTPI32*TAYAA + CTPI33*TAZAA
      VXECI = CTPI11*VXTP + CTPI12*VYTP + CTPI13*VZTP
      VYECI = CTPI21*VXTP + CTPI22*VYTP + CTPI23*VZTP
      VZECI = CTPI31*VXTP + CTPI32*VYTP + CTPI33*VZTP
      XECI = CTPI11*XTP + CTPI12*YTP + CTPI13*(ZTP - ZS)
      YECI = CTPI21*XTP + CTPI22*YTP + CTPI23*(ZTP - ZS)
      ZECI = CTPI31*XTP + CTPI32*YTP + CTPI33*(ZTP - ZS)
      RETURN
      END
C
C
      SUBROUTINE G6I
      RETURN
      END
C
C
      SUBROUTINE G6
      RETURN
      END
C
C
      SUBROUTINE S1I
C
C     HOMING SEEKER INITIALIZATION MODULE  S1I
C
      COMMON C(3415)
      DIMENSION  IPL(100)
      EQUIVALENCE (C(0317), FLGT),
      EQUIVALENCE (C(0347), FLGTS)
      EQUIVALENCE (C(3315), N )
      EQUIVALENCE (C(3316), IPL(1))
```

```
      IPL(N)=262
      IPL(N+1)=266
      IPL(N+2)=270
      IPL(N+3)=274
      IPL(N+4)=322
      IPL(N+5)=326
      IPL(N+6)=416
      IPL(N+7)=420
      IPL(N+8)=424
      IPL(N+9)=428
      N=N+10
      FLGT=0.0
      FLGTS=0.0
      RETURN
      END
C
C
C
      SUBROUTINE S1
C
C      HOMING SEEKER MODULE  S1
C
      COMMON C(3415)
      EQUIVALENCE (C(187), CNCIS1)
      EQUIVALENCE (C(197), CNDIS2)
      EQUIVALENCE  (C(212) , P)
      EQUIVALENCE  (C(216) , Q)
      EQUIVALENCE  (C(220) , R)
      EQUIVALENCE (C(0262), CMGYD)
      EQUIVALENCE (C(0265), CMGY)
      EQUIVALENCE (C(0266), CMGZD)
      EQUIVALENCE (C(0269), CMGZ)
      EQUIVALENCE  (C(270) , THTSD)
      EQUIVALENCE  (C(273) , THTS)
      EQUIVALENCE  (C(274) , PSISD)
      EQUIVALENCE  (C(277) , PSIS)
      EQUIVALENCE (C(0279), CMGYS)
      EQUIVALENCE (C(0280) , ANGLT)
      EQUIVALENCE (C(0316), RS)
      EQUIVALENCE (C(0317), FLGT)
      EQUIVALENCE (C(0318), RMT)
      EQUIVALENCE (C(0319), XDT)
      EQUIVALENCE (C(0320), YDT)
      EQUIVALENCE (C(0321), ZDT)
      EQUIVALENCE (C(322), CMZC1D)
      EQUIVALENCE (C(325), CMZC1)
      EQUIVALENCE (C(326), CMYC1D)
      EQUIVALENCE (C(329), CMYC1)
```

```
      EQUIVALENCE (C(335), FLAGS)
      EQUIVALENCE (C(0336), FLGD)
      EQUIVALENCE (C(337), CMGYC)
      EQUIVALENCE (C(338), CMGZC)
      EQUIVALENCE (C(0342), RMA)
      EQUIVALENCE (C(0343), XDA)
      EQUIVALENCE (C(0344), YDA)
      EQUIVALENCE (C(0345), ZDA)
      EQUIVALENCE (C(0346), CMGC)
      EQUIVALENCE (C(0347), FLGTS)
      EQUIVALENCE (C(0348), ERSAZ)
      EQUIVALENCE (C(0349), ERSEL)
      EQUIVALENCE (C(0356), CTHTS)
      EQUIVALENCE (C(0357), STHTS)
      EQUIVALENCE (C(0358), CPSIS)
      EQUIVALENCE (C(0359), SPSIS)
      EQUIVALENCE (C(0416), D1YD)
      EQUIVALENCE (C(0419), D1Y)
      EQUIVALENCE (C(0420), D1ZD)
      EQUIVALENCE (C(0423), D1Z)
      EQUIVALENCE (C(0424), D2YD)
      EQUIVALENCE (C(0427), D2Y)
      EQUIVALENCE (C(0428), D2ZD)
      EQUIVALENCE (C(0431), D2Z)
      EQUIVALENCE  (C(932) , T)
      EQUIVALENCE  (C(946) , TLC)
      DATA AKS/35.0/
      RAD=57.2957795
      IF (T.LT.TLC) GO TO 200
C     GIMBAL ANGLE LIMITS
      GMBLIM=14./RAD
      RXZA=SQRT(XDA**2+ZDA**2)
C
C     TRACKING RATE LIMIT
      TRLIM=20./RAD
      IF (RMA.LE.RS) GO TO 10
      FLAGS=0.0
C     FLAGS SIGNALS START OF SEARCH
C     FLGT SIGNALS SEEKER LOCKON AFTER SEARCH
      FLGT=0.0
C     FLGTS SIGNALS END OF SEARCH
      FLGTS=0.0
C     FLGD SIGNALS DETECTION OF TARGET
      FLGD=0.0
C     FLAGLT SIGNALS TARGET OUTSIDE SEEKER FOV
      FLAGLT=0.0
C     SEEKER POINTING ERRORS-ATIGS TARGET
      ELA=ATAN(-ZDA/XDA)
```

```
      AZA=ATAN(YCA/RXZA)
C     KEEP SEEKER CAGED IF POINTING ERRORS EXCEED GIMBAL LIMITS
      ELA1=ELA
      AZA1=AZA-4./RAD
      IF (ABS(ELA1).GT.GMBLIM) GO TO 200
      IF (ABS(AZA1).GT.GMBLIM) GO TO 200
C     PRECESS SEEKER TO SEARCH INITIATION PT
      CMGYC=AKS*ELA1
      CMGZC=AKS*AZA1
      GO TO 90
C    SEARCH SEQUENCE
   10 IF (FLAGS.GT.0.0) GO TO 20
      TME=T
      FLAGS=1.0
      WRITE (6,150)  RMA,T
   20 IF (FLGTS.GT.0.0) GO TO 75
      TS = T-TME
      IF (TS.GT.0.4) GO TO 30
      CMGYC=-10./RAD
      CMGZC=0.
      CMGC=ABS(CMGYC)
      GO TO 90
   30 IF (TS.GT.1.35) GO TO 40
      CMGYC=0.
      CMGZC=10./RAD
      CMGC=ABS(CMGZC)
      GO TO 90
   40 IF (TS.GT.1.775) GO TO 50
      CMGYC=TRLIM
      CMGZC=0.
      CMGC=ABS(CMGYC)
      GO TO 90
   50 IF (TS.GT.2.4) GO TO 60
      CMGYC=0.
      CMGZC=-TRLIM
      CMGC=ABS(CMGZC)
      GO TO 90
   60 IF (TS.GT.2.625) GO TO 65
      CMGYC=-TRLIM
      CMGZC=0.
      CMGC=ABS(CMGYC)
      GO TO 90
   65 IF (TS.GT.3.425) GO TO 70
      CMGYC=0.
      CMGZC=TRLIM
      CMGC=ABS(CMGZC)
      GO TO 90
C     END OF SEARCH
```

```
   70 IF (FLGE.GT.0.0) GO TC 90
      CMGZC=C.C
      CMGYC=C.C
      WRITE (6,130)
      FLGTS=1.C
      GO TC SC
   75 IF (FLGE.GT.0.0) GC TC 80
      GO TC SC
C  8C REPCSITICN SEEKER ON CETECTED TARGET
   8C IF (FLGT.GT.0.0) GC TC 85
      RYZT=SCRT(YDT**2+ZDT**2)
      CMGYC=-TRLIM*ZDT/RYZT
      CMGZC=TRLIM*YDT/RYZT
      IF (ANGLT.LE.0.25) FLGT=1.0
      GO TC SC
C    RADIOMETER ANC SENSCR FILTER LAGS
   85 WS=201.
      WSF=125.7
      D1YD=WS*(.2*ERSEL-D1Y)
      D1ZD=WS*(.2*ERSAZ-D1Z)
C     RECEIVER NCISE MODEL
      SGMNOI=3.C4815E-05*RMT*2.718282**(RMT*5.614624E-06)
      ELNCIS=0.2*SGMNOI*CNCIS1
      AZNOIS=C.2*SGMNOI*CNCIS2
C     ADC NCISE TO FILTEREC RACIOMETER SIGNAL
      D1YN=D1Y+ELNCIS
      D1ZN=D1Z+AZNOIS
      CMYC1D=WSF*(AKS*D1YN/RAC-OMYC1)
      OMZC1D=WSF*(AKS*D1ZN/RAC-OMZC1)
      CMGZC=OMZC1
      CMGYC=CMYC1
C     IF SEEKER LCSES TGT,PRECESSICN COMMANCS GO TC ZERO
      IF (ABS(ANGLT).LE.2.5) GC TO 90
   88 IF(FLAGLT.GT.0.0)  GC TC 90
      WRITE (6,140) ANGLT
      FLAGLT=1.0
C     SEEKER PRECESSION RATES
   9C IF (ABS(CMGYC).GE.TRLIM)  OMGYC=SIGN(TRLIM,CMGYC)
      IF (ABS(CMGZC).GE.TRLIM)  OMGZC=SIGN(TRLIM,CMGZC)
C    RESOLVE GIMBAL Y-AXIS ANG RATE TO SEEKER Y-AXIS
      OMGYS=CMGY*CPSIS+SPSIS*(R*STHTS-P*CTHTS)
C    SEEKER CYNAMICS
      D2YD=2.4*(CMGYC-OMGYS)
      CMGYC=2139.*(D2Y+D2YC/64.3)
      D2ZD=4.4*(CMGZC-OMGZ)
      CMGZD=1445.925*(D2Z+D2ZD/64.3)
C     SEEKER EULER ANGLE RATES
      THTSC=CMGY-C
```

```
      PSISD=CAGZ-P*STHTS-R*CTHTS
      THTLIM=SIGN(GMBLIM,THTS)
      PSILIM=SIGN(GMBLIM,PSIS)
      IF(ABS(THTS).GT.GMBLIM)  THTSD=50.*(THTLIM-THTS)
      IF(ABS(PSIS).GT.GMBLIM)  PSISD=50.*(PSILIM-PSIS)
  200 RETURN
 130  FORMAT(1H,10X,'END OF SEARCH,NO TARGET DETECTION')
  140 FORMAT (1H,10X,'SEEKER LOST TARGET,LOOK ANGLE=',F8.3)
  150 FORMAT (1H,10X,'START SEARCH,ATIGS TGT RGE=',F10.3,3X,'T=',F8.3)
      END
C
C
C
      SUBROUTINE S2I
C
      COMMON C(3415)
      EQUIVALENCE (C(0336), FLGD)
      EQUIVALENCE (C(0347),FLGTS)
      FLGD=0.0
      FLGTS=0.0
      RETURN
      END
C
C
      SUBROUTINE S2
C
C    RADIOMETER MODULE  S2
C
      COMMON C(3415)
      EQUIVALENCE (C(0172), BTA02)
      EQUIVALENCE (C(0224), A11)
      EQUIVALENCE (C(0228), A12)
      EQUIVALENCE (C(0232), A13)
      EQUIVALENCE (C(0236), A21)
      EQUIVALENCE (C(0240), A22)
      EQUIVALENCE (C(0244), A23)
      EQUIVALENCE (C(0248), A31)
      EQUIVALENCE (C(0252), A32)
      EQUIVALENCE (C(0256), A33)
      EQUIVALENCE (C(0273), THTS)
      EQUIVALENCE (C(0277), PSIS)
      EQUIVALENCE (C(0280), ANGLT)
      EQUIVALENCE (C(0290), XTP)
      EQUIVALENCE (C(0298), YTP)
      EQUIVALENCE (C(0306), ZTP)
      EQUIVALENCE (C(0310), XTTP)
      EQUIVALENCE (C(0311), YTTP)
      EQUIVALENCE (C(0312), ZTTP)
```

```
      EQUIVALENCE (C(0313), XATP)
      EQUIVALENCE (C(0314), YATP)
      EQUIVALENCE (C(0315), ZATP)
      EQUIVALENCE (C(0316), RS)
      EQUIVALENCE (C(0317), FLGT)
      EQUIVALENCE (C(0318), RMT)
      EQUIVALENCE (C(0319), XDT)
      EQUIVALENCE (C(0320), YDT)
      EQUIVALENCE (C(0321), ZDT)
      EQUIVALENCE (C(335), FLAGS)
      EQUIVALENCE (C(0336), FLGD)
      EQUIVALENCE (C(0339), X2)
      EQUIVALENCE (C(0340), Y2)
      EQUIVALENCE (C(0341), Z2)
      EQUIVALENCE (C(0342), RMA)
      EQUIVALENCE (C(0343), XDA)
      EQUIVALENCE (C(0344), YDA)
      EQUIVALENCE (C(0345), ZDA)
      EQUIVALENCE (C(0346), OMGC)
      EQUIVALENCE (C(0347), FLGTS)
      EQUIVALENCE (C(0348), ERSAZ)
      EQUIVALENCE (C(0349), ERSEL)
      EQUIVALENCE (C(0356), CTHTS)
      EQUIVALENCE (C(0357), STHTS)
      EQUIVALENCE (C(0358), CPSIS)
      EQUIVALENCE (C(0359), SPSIS)
      EQUIVALENCE (C(0932), T)
      EQUIVALENCE (C(0946), TUC)
      RAD=57.2957795
      X1=XTTP-XTP
      Y1=YTTP-YTP
      Z1=ZTTP-ZTP
      X2=XATP-XTP
      Y2=YATP-YTP
      Z2=ZATP-ZTP
C     TARGET POSITION IN BODY AXIS SYSTEM
      XMT=A11*X1+A12*Y1+A13*Z1
      YMT=A21*X1+A22*Y1+A23*Z1
      ZMT=A31*X1+A32*Y1+A33*Z1
      RMT=SQRT(XMT**2+YMT**2+ZMT**2)
C     ATIGS TGT POSITION IN BODY AXIS SYSTEM
      XMA=A11*X2+A12*Y2+A13*Z2
      YMA=A21*X2+A22*Y2+A23*Z2
      ZMA=A31*X2+A32*Y2+A33*Z2
      RMA=SQRT(XMA**2+YMA**2+ZMA**2)
      IF (T.LT.TUC) GO TO 200
C     SEEKER DIRECTION COSINES WRT BODY (PITCH-YAW SEQUENCE)
      CTHTS=COS(THTS)
```

```
      STHTS=SIN(THTS)
      CPSIS=COS(PSIS)
      SPSIS=SIN(PSIS)
      B11=CPSIS*CTHTS
      B12=SPSIS
      B13=-CPSIS*STHTS
      B21=-SPSIS*CTHTS
      B22=CPSIS
      B23=SPSIS*STHTS
      B31=STHTS
      B32=0.
      B33=CTHTS
C     RESOLVE TGT POSITION FROM BODY SYSTEM TO SEEKER AXES
      XDT=B11*XMT+B12*YMT+B13*ZMT
      YDT=B21*XMT+B22*YMT+B23*ZMT
      ZDT=B31*XMT+B32*YMT+B33*ZMT
      RYZT=SQRT(YDT**2+ZDT**2)
      RXYT=SQRT(XDT**2+YDT**2)
C     RESOLVE ATIGS TGT POSITION FROM BODY SYSTEM TO SEEKER AXES
      XDA=B11*XMA+B12*YMA+B13*ZMA
      YDA=B21*XMA+B22*YMA+B23*ZMA
      ZDA=B31*XMA+B32*YMA+B33*ZMA
      IF (FLAGS.GT.0.0) GO TO 5
      FLG1=0.0
      GO TO 200
    5 ANGLTR=ATAN(RYZT/XDT)
      ANGLT=ANGLTR*RAD
      IF (FLGTS.GT.0.0) GO TO 30
C     BETA=EFFECTIVE SEARCH BEAM WIDTH (DEG)
      BETA=6.2-7.07E-09*SQRT(CMGC*RAD/20.)*RMA**2
      BTAD2=BETA/2.0
C     TEST FOR TARGET DETECTION
      IF (FLG1.GT.0.0) GO TO 10
C     FLG1 SET WHEN TARGET ENTERS BEAM
      IF (ABS(ANGLT).LE.BTAD2) FLG1=1.0
      GO TO 200
   10 IF (FLGD.GE.1.0) GO TO 20
      TME=T
C     FLGD SET WHEN TARGET LEAVES BEAM
      IF (ABS(ANGLT).GE.BTAD2) FLGD=1.0
      GO TO 200
   20 TD=T-TME
C     DETECTION DECLARED 30 MSEC AFTER TGT LEAVES BEAM
      IF (TD.GE.0.03) GO TO 25
      GO TO 200
   25 IF (FLGTS.GE.1.0) GO TO 30
      WRITE (6,160) T
      FLGTS=1.0
```

```
      GO TO 200
   30 IF (FLGT.LE.0.0) GO TO 200
C     RADIOMETER ERROR SIGNAL
      CHI1=2.0193*(2.*ANGLT/3.-1.0)
      CHI2=2.0193*(2.*ANGLT/3.+1.0)
      IF (CHI1.EQ.0.0) GO TO 40
      ERSIG=(SIN(CHI1)/CHI1)**2-(SIN(CHI2)/CHI2)**2
      GO TO 50
   40 ERSIG=1.0-(SIN(CHI2)/CHI2)**2
C     ERSIG=ZERO IF SEEKER LOSES TRACK
   50 IF (ANGLT.GT.2.5)  ERSIG=0.0
C     RESOLVE INTO AZ,EL ERROR SIGNALS (SMALL ANGLE APPROX)
      IF (RYZT.LE.0.0) GO TO 60
      ERSAZ=ERSIG*YDT/RYZT
      ERSEL=ERSIG*(-ZDT)/RYZT
      GO TO 200
   60 ERSAZ=0.0
      ERSEL=0.0
  200 RETURN
 160  FORMAT (1H,10X,'END OF SEARCH,TARGET DETECTED.T=',F8.3)
      END
C
C
C
      SUBROUTINE S3I
      RETURN
      END
C
C
      SUBROUTINE S3
      RETURN
      END
C
C
      SUBROUTINE S4I
      RETURN
      END
C
C
      SUBROUTINE S4
      RETURN
      END
C
C
      SUBROUTINE S5I
      RETURN
      END
C
```

```
C
C
      SUBROUTINE S5
C
C  ACCELEROMETERS AND GYROS MODULE
C
      COMMON C(3415)
      EQUIVALENCE (C(0136),CG)
      EQUIVALENCE (C(0212),P)
      EQUIVALENCE (C(0213),QD)
      EQUIVALENCE (C(0216),Q)
      EQUIVALENCE (C(0217),RD)
      EQUIVALENCE (C(0220),R)
      EQUIVALENCE (C(710), AX)
      EQUIVALENCE (C(0711),AY)
      EQUIVALENCE (C(0712),AZ)
      EQUIVALENCE (C(1644),AZM)
      EQUIVALENCE (C(1648),AYM)
      EQUIVALENCE (C(1652),QM)
      EQUIVALENCE (C(1656),RM)
      EQUIVALENCE (C(0980), FM)
      EQUIVALENCE (C(1749),XAC)
      DATA FLGA/0.0/
C  ACCELEROMETER OFFSET FROM CG
C     ACCELEROMETERS AT STA. 134.5
      XACC=CG-4.9433
      AX1=AX-XACC*(R**2+Q**2)
      AZ1=AZ+XACC*(R*P-QD)
      AY1=AY+XACC*(Q*P+RD)
      IF (FLGA.GT.0.0) GO TO 10
C     IDEAL ACCELEROMETERS
      AXM=AX1
      AYM=AY1
      AZM=AZ1
      GO TO 20
C     DIGITAL ACCELEROMETERS
C       TRANSFORMATION TO SKEWED ACCELEROMETER AXES
   10 BA11=.57735027
      BA12=BA11
      BA13=BA11
      BA21=-.70710678
      BA22=-BA21
      BA23=0.
      BA31=BA11*BA21
      BA32=BA31
      BA33=-2.0*BA31
      AXA=BA11*AX1+BA12*AY1+BA13*AZ1
      AYA=BA21*AX1+BA22*AY1+BA23*AZ1
```

```
      AZA=EA31*AX1+BA32*AY1+BA33*AZ1
C     ACCELERCMETER QUANT. STEF=.194 G
      AQS=6.24176
      ANSX=AXA/ACS
      AXMA=ACS*AINT(ANSX)
      ANSY=AYA/ACS
      AYMA=ACS*AINT(ANSY)
      ANSZ=AZA/ACS
      AZMA=ACS*AINT(ANSZ)
C      TRANSFCRMATION BACK TC ECDY AXES
      AXM=BA11*AXMA+BA21*AYMA+BA31*AZMA
      AYM=BA12*AXMA+BA22*AYMA+EA32*AZMA
      AZM=BA13*AXMA+BA23*AYMA+EA33*AZMA
C ***** ICEAL GYRCS *****
   20 CM=C
      RM=R
      FM=P
      RETURN
      END
C
C
      SUBRCUTINE S6I
      RETURN
      ENC
C
C
      SUBRCUTINE S6
      RETURN
      ENC
C
C
      SUBRCUTINE S7I
      RETURN
      END
C
C
      SUBRCUTINE S7
      RETURN
      ENC
C
C
      SUBRCUTINE S8I
      RETURN
      ENC
C
C
      SUBRCUTINE S8
      RETURN
```

```
      END
C
C
      SUBROUTINE SSI
      RETURN
      END
C
C
      SUBROUTINE SS
      RETURN
      END
C
C
      SUBROUTINE S10I
      RETURN
      END
C
C
      SUBROUTINE S10
      RETURN
      END
C
C
C
      SUBROUTINE BOOST (TT,THRUST,FF)
C
      DIMENSION T(25),F(25),W(25)
      COMMON C(3415)
      EQUIVALENCE (C(0933),T1)
      EQUIVALENCE (C(0507), F)
C
      DATA T(1)/0.0/,T(2)/.075/,T(3)/.08/,T(4)/.10/,T(5)/.50/,T(6)/1.0/,
     1T(7)/1.50/,T(8)/1.75/,T(9)/1.95/,T(10)/2.125/,T(11)/3.125/,
     2T(12)/3.505/,T(13)/3.70/,T(14)/3.80/,T(15)/3.875/,T(16)/3.95/,
     3T(17)/4.025/,T(18)/4.075/,T(19)/4.15/,T(20)/4.20/,T(21)/4.25/,
     4T(22)/4.35/,T(23)/4.475/,T(24)/4.60/,T(25)/4.75/
C
      DATA F(1)/0.0/,F(2)/2000./,F(3)/16000./,F(4)/17800./,F(5)/19200./,
     1F(6)/21500./,F(7)/24100./,F(8)/25100./,F(9)/25800./,F(10)/26100./,
     2F(11)/26500./,F(12)/26500./,F(13)/26400./,F(14)/25600./,
     3F(15)/24800./,F(16)/24000./,F(17)/22000./,F(18)/16000./,
     4F(19)/12000./,F(20)/9000./,F(21)/6900./,F(22)/4000./,F(23)/1900./,
     5F(24)/900./,F(25)/0.0/
C
      DATA W(1)/0.0/,W(2)/.321/,W(3)/2.841/,W(4)/6.575/,W(5)/61.178/,
     1W(6)/111.759/,W(7)/148.870/,W(8)/172.301/,W(9)/182.143/,
     2W(10)/206.751/,W(11)/255.867/,W(12)/362.863/,W(13)/376.037/,
     3W(14)/381.912/,W(15)/388.351/,W(16)/392.396/,W(17)/397.734/,
```

```
     4W(18)/401.847/,W(19)/404.344/,W(20)/406.206/,W(21)/407.368/,
     5W(22)/408.396/,W(23)/409.175/,W(24)/409.883/,W(25)/410.296/
C
      TIME=TT-T1
      IF(TIME.GT.0.) GO TO 3
      FF = 0.0
      THRUST = 0.0
      GO TO 4
    3 DO 1 I = 2,25
      IF(T(I).GE.TIME) GO TO 2
    1 CONTINUE
      CALL EXIT
    2 AK=(TIME-T(I-1))/(T(I)-T(I-1))
      THRUST = F(I-1) + (F(I) - F(I-1))*AK
      FF = W(I-1) + (W(I) - W(I-1))*AK
       PSF=2116.217*((1.0-(6.87534703E-06)*H)**5.25627005)
      THRUST=THRUST+(2116.2-PSF)*.715428
    4 RETURN
      END
C
C
C
      SUBROUTINE RAMJET (HIN,XMIN,AIN,WFIN,XIN,CT)
C
C  SIMPLIFIED RAMJET MODEL
C   USES TABLE LOOK-UP FOR THRUST COEFF AND FUEL FLOW RATE
C
      DIMENSION TM1(6),TM2(7),TH(4),TETA(5),TCF1(6,5,4),TCF2(7,5,4),
     1TWF(6,5,4)
      COMMON C(34151)
C
      DATA TM1/2.3,2.5,2.7,2.9,3.1,3.3/
C
      DATA TM2/0.5,1.0,1.5,2.0,2.5,3.0,3.3/
C
      DATA TH/500.,13000.,20000.,35000./
C
      DATA TETA/0.,2.,4.,6.,8./
C
      DATA TCF1/ .3814,.2958,.2075,.1374,
     A.0844,.0443, .3794,.2843,.2025,.1339,.0805,.0402, .3731,.2818,
     B.1915,.1203,.0666,.0258, .3476,.2526,.1699,.1049,.0555,.0173,
     C .3251,.2355,.1595,.1001,.0549,.0201,  .4357,.3412,.2453,.1651,
     D.1115,.0662, .4337,.3293,.2401,.1654,.1075,.0639, .4273,.3268,
     E.2287,.1512,.0927,.0485, .4012,.2967,.2061,.1350,.0810,.0396,
     F .3782,.2791,.1954,.1300,.0805,.0425,  .4944,.386,.2798,.1963,
     G.1341,.088, .4924,.373,.2742,.1924,.1296,.0832, .4857,.3706,.2618,
     H.1766,.1132,.066, .4573,.3376,.237,.1588,.1003,.056, .4328,.3186,
```

```
     I.2254,.1535,.0999,.0595,  .6273,.5139,.3961,.2997,.2243,.1661,
     J .625,.5028,.3911,.296,.2202,.1616, .6179,.4598,.3801,.2821,.2054,
     K.1459, .5521,.4721,.3588,.2662,.1936,.1367, .5708,.4555,.3484,
     L.2613,.1932,.1398/
C
      DATA TCF2/-.5727,-.3661,-.4238,
     A-.2615,-.2067,-.2074,-.2091, -.5727,-.3658,-.4235,-.2618,-.2067,
     B-.2072,-.209, -.5728,-.3655,-.4233,-.2623,-.2065,-.2069,-.2089,
     C -.5729,-.3631,-.4218,-.2635,-.2064,-.2065,-.2086, -.5731,-.3618,
     D-.4205,-.2648,-.2062,-.2061,-.2081,  -.5692,-.3649,-.4226,-.2603,
     E-.2055,-.2060,-.2075, -.5693,-.3646,-.4223,-.2606,-.2054,-.2058,
     F-.2074, -.5693,-.3643,-.4221,-.2612,-.2053,-.2055,-.2073, -.5695,
     G-.3619,-.4207,-.2624,-.2052,-.2051,-.207, -.5697,-.3606,-.4194,
     H-.2636,-.205,-.2047,-.2065,  -.5659,-.3637,-.4212,-.259,-.204,
     I-.2041,-.2054, -.5659,-.3633,-.421,-.2593,-.2039,-.2039,-.2052,
     J -.566,-.3631,-.4208,-.2598,-.2038,-.2036,-.2052, -.5662,-.3606,
     K-.4154,-.261,-.2037,-.2032,-.2048, -.5664,-.3594,-.4181,-.2622,
     L-.2035,-.2028,-.2044,  -.5553,-.3602,-.4186,-.2569,-.2027,-.203,
     M-.2043, -.5554,-.3599,-.4184,-.2572,-.2026,-.2028,-.2042, -.5555,
     N-.3596,-.4182,-.2577,-.2025,-.2025,-.2042, -.5557,-.3572,-.4168,
     O-.2589,-.2023,-.2021,-.2038, -.5559,-.356,-.4156,-.2601,-.2021,
     P-.2017,-.2033/
C
      DATA TWF/ 2.95,2.95,2.885,2.835,
     A2.806,2.806, 2.941,2.882,2.853,2.805,2.771,2.763, 2.911,2.867,
     B2.779,2.703,2.65,2.619, 2.778,2.7,2.64,2.588,2.552,2.532, 2.665,
     C2.604,2.571,2.549,2.542,2.553,  1.966,1.966,1.927,1.894,1.877,
     D1.876, 1.961,1.924,1.906,1.876,1.855,1.849, 1.942,1.914,1.86,
     E1.812,1.779,1.76, 1.859,1.81,1.773,1.74,1.718,1.706, 1.788,1.751,
     F1.73,1.716,1.712,1.718,  1.561,1.561,1.532,1.507,1.494,1.493,
     G 1.557,1.529,1.516,1.493,1.477,1.473, 1.543,1.522,1.481,1.445,
     H1.42,1.406, 1.48,1.444,1.416,1.391,1.375,1.365, 1.427,1.399,1.383,
     I1.373,1.37,1.375,  .968,.968,.952,.94,.933,.933, .965,.951,.944,
     J.933,.924,.922, .958,.947,.926,.908,.895,.888, .926,.907,.893,.88,
     K.872,.867, .899,.884,.876,.871,.869,.872/
C
      H=HIN*3.280839895
      IF (XIN.GT.0.0) GO TO 10
C     RAMJET-ON CRUISE
      CT=THRECL(XMIN,AIN,H,TM1,TETA,TH,TCF1,6,5,4)
      WFIN=THRECL(XMIN,AIN,H,TM1,TETA,TH,TWF,6,5,4)
      GO TO 20
C     RAMJET-OFF CRUISE
   10 CT=THRECL(XMIN,AIN,H,TM2,TETA,TH,TCF2,7,5,4)
      WFIN=0.0
   20 RETURN
      END
C
```

```
C
C
      SUBROUTINE ENGINE (HIN,XMIN,AIN,WFIN,TIN,XIN,YIN,ZIN,CT,SMARG,
     1                   IFI,IREAD)
C
C
C
C     NWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
C     CPIA NOMENCLATURE
C
      REAL ISP,ISTAR,M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
      REAL MACR,MACRS,MACH
      INTEGER COUNT
C
      COMMON/CBLS/  ENG(30),IENG(15)
      COMMON /RJ/ A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
      COMMON /RJ/ ACR,BLEEC
      COMMON /RJ/ CCASUB,CCASUP,COB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
      COMMON /RJ/ ER,ERRL,ERLL,ETAC,ETAN,F
      COMMON /RJ/ FAR,G,GAMMA,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA6
      COMMON /RJ/ H,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
      COMMON /RJ/ COUNT,ICONV,ICONVP,IFIRST
      COMMON /RJ/ M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
      COMMON /RJ/ MACR,MACRS,PM
      COMMON /RJ/ P,P1,P2,P3,P4,P5,P6,PT30I,PRAR
      COMMON /RJ/ PT,PT1,PT2,PT3,PT4,PT5,PT6,PMOPT,PMOPT1,PMOPT2,PMOPT3
      COMMON /RJ/ PHI1,PHI2,PHI3,PHI4,PHI5,PHI6,PHIPRI,Q
      COMMON /RJ/ RADEG,R,SA,SART,SQPTOT
      COMMON /RJ/ T,T1,T2,T3,T4,T5,T6,TIME
      COMMON /RJ/ TT,TT1,TT2,TT3,TT4,TT5,TT6
      COMMON /RJ/ V,V1,V2,V3,V4,V5,V6
      COMMON /RJ/ WA,WF,X,Y,Z
C
C     *     *     *     *     *     *     *
C
      COMMON /EXHALS/ TABLE(1000)
C
      COMMON /WRITCT/ WRIT(20)
C
      NAMELIST /MCNT/ ENG,IENG
C
C      *** THE INFORMATION BELOW NOTES HOW THE NAMELIST
C      DATA IS INPUT AND WHAT THE INPUTS ARE ***
C
C     ENG(1)=A1/AR OR A1 (M SQ)  SEE I6
C     ENG(2)=A3/AR OR A3 (M SQ)  SEE I6
C     ENG(3)=A4/AR OR A4 (M SQ)  SEE I6
C     ENG(4)=A5/AR OR A5 (M SQ)  SEE I6
```

```
C      ENG(5)=AE/AR CR A6 (M SQ)  SEE I6
C      ENG(6)=AR (METRES SQUARED)
C      ENG(7)=CDE  BURNER DRAG COEFFICIENT
C      ENG(8)=ETAN, NOZZLE THRUST EFFICIENCY
C      ENG(9)=ETAC, COMBUSTION EFFICIENCY
C      ENG(10)=WF,  FUEL FLOW RATE (KG/SEC) SEE I5
C      ENG(11)=ISP STAR (FOR GAS GENERATOR SYSTEMS)
C      ENG(12)=PHI OF PRIMARY (FOR GAS GEN SYSTEMS)
C      ENG(13)=BLEED FLOW PERCENTAGE/ 100.
C      ENG(14)=AIR TO FUEL STOICHEMETRIC (AFS)
C      ENG(15)=CAM, NOZZLE MASS FLOW COEFFICIENT
C
       REAL MCOSE
       DIMENSION  MCOSE(176), EAGLE(176)
C
C *** THE INPUTS BELOW ARE SWITCHES ***
C      IENG(1)=I1, ENGINE TYPE SWITCH
C        IF I1=0,RAMJET TYPE  IF I1=1, GAS GEN TYPE
C      IENG(2)=I2, COMBUSTION EFF SWITCH
C        IF I2=0,ETAC IS SET BY INPUT
C        IF I2=1,ETAC IS SET IN A SUBROUTINE
C      IENG(3)=I3, ATMOSPHERE CHOICE SELECTION, SEE MEMO BY RGL
C         I3=2 IS STANDARD DAY
C      IENG(4)=I4, REAL AIR SWITCH
C        IF I4=0, USE THE REAL AIR SOLUTION
C        IF I4=1, USE THE GAMMA=1.4 SOLUTION
C      IENG(5)=I5, WF OR ER SOLUTION SWITCH
C        IF I5=1, THE PROGRAM USES ER IN THE SOLUTION
C        IF I5=2, THE PROGRAM USES WF IN THE SOLUTION
C       IENG(6)=I6, INLET DATA TYPE SWITCH
C         IF I6=0, THE INPUTS ARE RATIOS BASED ON AR
C         IF I6=1, THE AREAS ARE ABSOLUTE VALUES
C        IENG(7)=I7, PRINT EACH ITERATION SWITCH
C          IF I7=0, DO NOT PRINT DEBUG DATA EACH ITERATION
C         IF I7.NE.0, DO PRINT DEBUG DATA
C         IENG(8)=I8, COMBUSTION PRESSURE EFFECT SWITCH
C           IF I8=0,THERE IS NO PRESSURE EFFECT COMPUTED
C           IF I8=1, THE PRESSURE EFFECT ON COMBUSTION TEMP IS COMPUTED
C
       EQUIVALENCE(TABLE(206),MCOSE(1))
       EQUIVALENCE(TABLE(382),EAGLE(1))
C
C *** EXAMPLE OF NAMELIST INPUT ***
C    $MONT
C    ENG=.5256,.65,.85,.44444,.96,.1140092,0.,.98,.95,2.,.50,1.2,.04,13.8
C    1.0,
C    IENG=0,0,2,0,1,0,0,0,
C    $END
```

```
C
      IF (IREAD.EQ.1) GO TO 10
C     COMMENT  READ IN DATA AND RETURN TO CALLING ROUTINE
      READ (5,MCNT)
      WRITE (6,MCNT)
      RETURN
C     COMMENT  PUT DATA IN FORM THAT THE ATHODYC PROGRAM CAN USE
   10 CONTINUE
      MACH=0.0
      H=HIN
      M=XMIN
      ALPHA=AIN
      WF=WFIN
      TIME=TIN
      IF(WFIN.LT.0.0001.AND.TIME.GT.20.) IFUEL=4
      IFIRST=IFI
C                         X=XIN           @  GAMMA
      X=XIN
C                         Y=YIN           @  NOT USED
      Y=YIN
C                         Z=ZIN           @  NOT USED
      Z=ZIN
      A1=ENG(1)
      IF(TIME.LT.10.0) A1=.43244
      IF(TIME.GT.10.0.AND.TIME.LT.50.0) A1=.43162+.00008639*TIME
      IF (TIME.GE.50.0) A1=.43594
      A3=ENG(2)
      A4=ENG(3)
      IFUEL=0
      IF(XIN.GE.1.0) IFUEL=5
      IF(TIME.LE.12.5) A5=.50326
      IF(TIME.GT.12.5.AND.TIME.LE.40.) A5=.51410+.00023839*TIME
      IF(TIME.GT.40..AND.TIME.LE.66.) A5=.51776+.00014694*TIME
      IF(TIME.GT.66.) A5=.52746
      A6=ENG(5)
      AR=ENG(6)
      CDB=ENG(7)
      ETAN=ENG(8)
      ETAC=ENG(9)
      IF (IENG(5).EQ.2) WF=ENG(10)
      ISTAR=ENG(11)
      PHIPRI=ENG(12)
      BLEED=ENG(13)
      AFS=ENG(14)
      CNM=ENG(15)
      IF (CNM.LT.0.5) CNM=1.0
      I1=IENG(1)
      I2=IENG(2)
```

```
      I3=IENG(3)
      I4=IENG(4)
      I5=IENG(5)
      I6=IENG(6)
      I7=IENG(7)
      I8=IENG(8)
      IF (I6.EQ.0) A1=A1*AR
      IF (I6.EQ.0) A3=A3*AR
      IF (I6.EQ.0) A4=A4*AR
      IF (I6.EQ.0) A5X=A5*AR
      IF (I6.EQ.0) A6=A6*AR
      IF (I6.EQ.1) A5X=A5
      A5=A5X*CNM
      IF (IFIRST.EQ.-1) GO TO 20
      GO TO 30
   20 CONTINUE
C     CMMENT  INITIALIZE ,SET CONSTANTS AND WRITE AREAS
C
C
C      AIRBREATHING GEOMETRY       ONLY CALLED ON FIRST PASS
C     CALCULATE GEOMETRY PARAMETERS
C
C
      A4OA5=A4/A5
      GAM=9.0/7.0
      M4=FMARL(GAM,A4OA5)
      PHI4=PHIM(GAM,M4)
      A6OA5=A6/A5
      M6=FMARH(GAM,A6OA5)
      PHI6=PHIM(GAM,M6)
C
C     IDENTIFY AND PRINT PARAMETERS
C
      WRITE (6,420)
      WRITE (6,430) A1
      WRITE (6,440) A3
      WRITE (6,450) A4
      WRITE (6,460) A5X
      WRITE (6,470) A6
      WRITE (6,480) AR
      WRITE (6,490) PHI4
      WRITE (6,500) PHI6
      WRITE (6,510) CDB
      WRITE (6,520) CNM
   30 CONTINUE
C
C     SET CONSTANTS AND INITIALIZE SUBROUTINES
C                           G IN KG M/SEC SQ - N
```

```
C
      RADEG=.572957779E+02
      G=9.80665
C
C     START CALCLLATIONS FCR EACH NEW POINT
C
      COUNT=0
      ICCNVP=1
C
C     NEW ITERATICNS START HERE
C
   40 ICCNV=0
C
C     CHECK TC SEE IF 100 ITERATIONS OF THE SAME PCINT HAVE BEEN MADE
C
      IF (CCUNT-100) 60,60,50
C
C     IF NCT CCNVERGED AFTER 100 ATTEMPTS, STOP
C
   50 WRITE (6,530)
C
      CALL EXIT
C     IF NCT CCNVERGED WITH LESS THAN 100 ATTEMPTS, TRY AGAIN
C
   60 CONTINUE
      IF(CCUNT.GT.89) I7=10
C     IF ABCVE STATEMENT NCT CCNVERG,PRINT LAST 10
      IF (COUNT.NE.0) GO TC 90
      IF (IFIRST.EC.-1) CALL AIR(H,M,-1,P,T,A,DEN,Q,G,1,1)
      IF (IFIRST.EC.-1) GC TC 70
      I3=I3+100
      CALL AIR(H,M,1,P,T,A,DEN,Q,G,1,1)
   70 G=9.80665
      IF (I4.EC.1) GO TO 80
C                           ENTHR IN CAL/GM
      ENTHR=46.427E+.23949*(T-194.444)
C                           V IN METRES PER SEC
      V=M*A
C                           ENERG IN CAL/GM
      ENERG=(V**2.)/8368.
      ENTHAL=ENTHR+ENERG
C                           ECGM IN CAL/GM
      ECGM=ENTHAL+ENERG
C                           TT3 IN DEGREE RANKINE
      TT3=539.31+(7.5164*ECGM)-(3.3559E-03*ECGM*ECGM)+(4.65976E-06*(ECGM
     1**3.))-(5.22956E-09*(ECGM**4.))
      TT3=TT3/1.8
      GAMMA3=1.4048-(6.09654E-04*ECGM)+(2.12102E-06*ECGM*ECGM)-(5.56849E
```

```
     1-05*(ECGM**3.))+(6.45543E-12*(ECGM**4.))
      GO TO 90
   80 CONTINUE
      GAMMA3=1.4
      TT3=T/FTCTT(1.4,M)
      T3=TT3*FTCTT(1.4,M3)
   90 CONTINUE
C
      IF (IFIRST) 100,110,110
  100 WRITE (6,540)
C
C     COMBUSTOR
C
      IF (I2.EQ.1) CALL ETA
      CALL TAE (AFS)
      V=-0.0
      WA=10.0
      M3=.2
      MACRS=1.
      CALL HCRA
C
C     CALCULATION
C
      WRITE (6,230)
C
C     DIFFUSER
C
      CALL HCF
      GO TO 190
  110 CONTINUE
      IF (M.GT.1.07) GO TO 130
C        ***** BEGIN SUBSONIC CALL SEQUENCE *****
      IF (WF.LT..001) GO TO 120
      PT=P/FFCPT(1.4,M)
      FMOPT=FFMCPT(1.4,M)
      TT=T/FTCTT(1.4,M)
      ACR=.27661+2.4721*M-4.527*M*M+2.3036*M*M*M
      AINF=A1*ACR*1.05
      WA=(AINF*PT*FMOPT)/SQRT(TT)
      CALL HCRA
  120 CONTINUE
      IF (WF.LT..001) ER=0.0
      FAR=ER/AFS
      CALL SUBSON (H,M,FAR,CF,WF,Q,AR)
      ICONV=0
      COUNT=0
      GO TO 225
C        *****  END OF SUBSONIC CALL SEQUENCE ****
```

```
  130 CONTINUE
C
C      ********  WE SET SA HERE ********
C
      IF (I5.EQ.1) GO TO 140
      IF (I5.EQ.2) GO TO 150
      WRITE (6,240)
C GO THIS ROUTE IF ER IS INPUT OR CHANGED IN HCRA
  140 CONTINUE
      CALL HCRA
      IF(IFUEL.EQ.5) ER=0.0
      FAR=ER/AFS
      WF=FAR*WA
      AFR=1./FAR
      GO TO 160
C GO THIS ROUTE IS WF IS INPUT OR SET IN HCRA
  150 CONTINUE
      CALL HCRA
      IF(IFUEL.EQ.5) WF=0.0
      FAR=WF/WA
      ER=FAR*AFS
      IF(FAR.EQ.0.0) GO TO 1400
      AFR=1.0/FAR
      GO TO 160
 1400 AFR=0.0
C
  160 CONTINUE
C
      TOTT3=FTOTT(GAMMA3,M3)
      T3=TT3*TOTT3
      SQRTOT=SQRT(TT3)
C
C     TO FIND AIR SPECIFIC IMPULSE
C
      IF(I10.EQ.0) GO TO 161
      CONTINUE
      GO TO 169
  161 CONTINUE
      CALL INTR20 (TT3,ER,TT4I,TABLE,1)
      CALL INTR20 (TT3,ER,GAMMA4,TABLE,2)
      CALL INTR20 (TT3,ER,MW4,TABLE,3)
C
C      COMPUTE THE EFFECT OF PRESSURE ON TEMPERATURE
C
       IF(I8.EQ.0) GO TO 162
      G1=GAMMA4+1.0
      G2=GAMMA4-1.0
      G3=G1/G2
```

```
      G4=2.*G1*((2./G1)**G3)
      G5=SQRT(G4)
      PT4=(SA*WA)/(G5*A5)
      PT4ATM=PT4/101360.
      IF(PT4ATM.LT. .2) PT4ATM=.2
      IF(PT4ATM.GT. 20.) PT4ATM=20.
      IF(TT3.LT.299.) GO TO 1615
      IF(ER.GT.1.5) GO TO 1615
      CTEMP=TC(TT3,ER,PT4ATM)
 1615 CONTINUE
      TT4I=TT4I+CTEMP
  162 CONTINUE
C
C  NOW FINISH COMBUSTOR COMPUTATIONS
C
      R=8314.34/MW4
      WAX=WA/.4535924
      THROAT=A5X/.09290304
      TTT=((TT3*1.8)/1000.)**2.
      BSV=(WAX*TTT)/THROAT
      IF (I2.EQ.1) CALL ETA
      TT4=ETAC*(TT4I-TT3)+TT3
      BBB=2.0*R*(1.0+GAMMA4)*TT4/GAMMA4
      BBB=AMAX1(400.,BBB)
      SA=(1.0+FAR)*SQRT(BBB)
      SART=SA/SQRTCT
  165 CONTINUE
C
C     CMMENT  MAKE CALCULATIONS TAKING VARIABLE NOZZLE GAMMA INTO EFFECT
C
      A4CA5=A4/A5
      M4=FMARL(GAMMA4,A4OA5)
      PHI4=PHIM(GAMMA4,M4)
C
C      ******** WE COMPUTE CONDITIONS AT STATION 3 HERE ********
C
      STRCG=23.9613
      IF (I1.EQ.1) GO TO 170
C     RAMJET COMPUTATIONS - FUEL ADDED IN THE INLETS
      SARTA=STRCG*SQRT((GAMMA3+1.)/GAMMA3)*(1.+FAR)
      PHI3=(PHI4*SART/SARTA)+((GAMMA3*CDB*M3*M3)/(2.*XMCIR3*SARTA))
      GO TO 180
C     GAS GENERATOR SYSTEM COMPUTATIONS
  170 SARTE=STRCG*SQRT((GAMMA3+1.)/GAMMA3)
      PHI3=(PHI4*SART/SARTB)-((PHIPRI*ISTAR*FAR)/(SARTB*SQRTOT))+((GAMMA
     13*CDB*M3*M3)/(2.*XMCIR3*SARTB))
  180 CONTINUE
      M3=FMPHIM(GAMMA3,PHI3)
```

```
      FMCFT3=FFMCFT(GAMMA3,M3)
      PMOPT=FFMCPT(1.4,M)
      XMCIF3=FMCIR(GAMMA3,M3)
C
C      ******** WE NOW CALL THE INLET ROUTINE TO GET WA ********
C
      CALL HCF
C
  190 CONTINUE
C
C      ******** DEBUG OUTPUT ********
C
      IF (I7.EQ.0) GO TO 200
  201 CONTINUE
      FK=F/1000.
      P1K=P1/1000.
      P2K=P2/1000.
      P3K=P3/1000.
      P4K=P4/1000.
      P5K=P5/1000.
      P6K=P6/1000.
      PTK=PT/1000.
      PT1K=PT1/1000.
      PT2K=PT2/1000.
      PT3K=PT3/1000.
      PT4K=PT4/1000.
      PT5K=PT5/1000.
      PT6K=PT6/1000.
      QK=Q/1000.
      WRITE (6,250) A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
      WRITE (6,260) ACR,BLEED
      WRITE (6,270) CDASUB,CDASUP,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
      WRITE (6,280) ER,ERRL,ERLL,ETAC,ETAN,F
      WRITE (6,290) FAR,G,GAMMA,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA
     16
      WRITE (6,300) H,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
      WRITE (6,310) COUNT,ICONV,ICONVP,IFIRST
      WRITE (6,320) M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
      WRITE (6,330) MACR,MACPS,PM
      WRITE (6,340) PK,P1K,P2K,P3K,P4K,P5K,P6K,PT3CI,PRAR
      WRITE (6,350) PTK,PT1K,PT2K,PT3K,PT4K,PT5K,PT6K,PMOPT,PMCPT1,PMCFT
     12,PMCPT3
      WRITE (6,360) PHI1,PHI2,PHI3,PHI4,PHI5,PHI6,PHIPRI,QK
      WRITE (6,370) RADEG,R,SA,SART,SQRTOT
      WRITE (6,380) T,T1,T2,T3,T4,T5,T6,TIME
      WRITE (6,390) TT,TT1,TT2,TT3,TT4,TT5,TT6
      WRITE (6,400) V,V1,V2,V3,V4,V5,V6
      WRITE (6,410) WA,WF,X,Y,Z
```

```
C
C
  200 CONTINUE
C       SET IFTS, INCREASE COUNT AND CALCULATE PARAMETERS NOT IN COMMON
C
        COUNT=COUNT+1
C
C       CHECK CONVERGENCE, ICONVF IS USED TO MAKE SOLUTION CONVERGE TWICE
C       IF CONVERGED, PRINT.  IF NOT, GO BACK AND TRY AGAIN
C
        IF (IFIRST.EQ.-1) GO TO 225
        IF (ICONV+ICONVP) 210,220,210
  210 ICONVF=ICONV
        GO TO 40
  220 CONTINUE
C        ******** WE NOW COMPUTE THRUST IF CONVERGED ********
C
        A6CA5=A6/A5
        M6=FMARH(GAMMA4,A6CA5)
        PHI6=PHIM(GAMMA4,M6)
        IF(I10.GE.5) GO TO 221
        CF6=(WA*SA*PHI6*ETAN-A6*P)/(AR*Q)
  221 CONTINUE
        CFINF=2.0*ACR*MACRS*A1/AR
        CALPHA=COS(ALPHA/RADEG)
        CF=CF6-(CFINF*CALPHA)+CFB+CFC
        F=CF*Q*AR
        IF(WF.EQ.0.0) GO TO 1300
        ISP=F/(WF*G)
        GO TO 1500
 1300 ISP=0.0
 1500 CONTINUE
        G1=GAMMA4+1.0
        G2=GAMMA4-1.0
        G3=G1/G2
        G4=2.*G1*((2./G1)**G3)
        G5=SQRT(G4)
        PT5=(SA*WA)/(G5*A5)
C
C        PREPARE DATA FOR TRANSFER TO MAIN PROGRAM
C
  225 CONTINUE
        FIN=F
        XMIN=M
        AIN=ALPHA
        WFIN=WF
        CDADD=CDASUB+CDASUP+CFT
        CT=CF-CDADD
```

```
      SMARG=FN
      IFI=IFIRST
      FCCUNT=CCUNT+.01
      WRIT(1)=PT3CI
      WRIT(2)=WA
      WRIT(3)=TT4
      WRIT(4)=MACR
      WRIT(5)=ACR
      WRIT(6)=ER
      WRIT(7)=ISP
      WRIT(8)=FCCUNT
      WRIT(9)=WF
      WRIT(10)=ETAC
      WRIT(11)=CT
      WRIT(12)=FN
      WRIT(13)=FCCUNT
      WRIT(14)=ESV
      IF(PT3.EC.C.C) GO TC 12CC
      WRIT(15)=PT5/PT3
 12CC CONTINUE
      WRIT(16)=PT5/1000.
      FCCUNT=FCCUNT
      RETURN
C
  230 FORMAT ('      COMPUTE PHI3')
  240 FORMAT ('      I5 IS CUT CF RANGE-WHY')
  250 FORMAT (1H1,F8.2,8F7.4,2F7.3,F7.2)
  26C FORMAT (2F8.5)
  27C FORMAT (1CF8.5)
  28C FORMAT (5F8.5,F10.3)
  29C FORMAT (9F8.5)
  30C FORMAT (F9.2,2F8.3,1CI2)
  310 FORMAT (4I4)
  320 FORMAT (7F7.4,7F8.4)
  330 FORMAT (2F9.5,F7.3)
  340 FORMAT (7F10.2,2F7.5)
  35C FORMAT (7F10.2,4F7.5)
  360 FORMAT (7F8.5,F10.2)
  37C FORMAT (F9.5,2F10.5,2F8.5)
  38C FORMAT (8F1C.3)
  390 FORMAT (7F10.3)
  4CC FORMAT (7F10.3)
  410 FORMAT (2F8.3,3F10.5)
  420 FORMAT (54H1   AIR BREATHING ENGINE GEOMETRY
  430 FORMAT (14H          A 1   = F9.7,9H SQ METRE)
  44C FORMAT (14H          A 3   = F9.7,9H SQ METRE)
  45C FORMAT (14H          A 4   = F9.7,9H SQ METRE)
  46C FORMAT (14H          A 5   = F9.7,9H SQ METRE)
```

```
  470 FORMAT (14H         A 6   = F9.7,9H SQ METRE)
  480 FORMAT (14H         A R   = F9.7,9H SQ METRE)
  490 FORMAT (28H         PHI4                  = F7.5)
  500 FORMAT (28H         PHI6                  = F7.5)
  510 FORMAT (28H         BURNER DRAG COEFF. = F6.2)
  520 FORMAT (28H         NOZZLE MASS COEFF. = F6.2)
  530 FORMAT (36H DID NOT CONVERGE AFTER 100 ATTEMPTS)
  540 FORMAT (47H0     RAMJET PROPULSION                              )
C
      END
C
C
C
      SUBROUTINE ETA
C
C
C     NWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
C     CPIA NOMENCLATURE
C
      REAL ISP,ISTAR,M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
      REAL MACR,MACRS
      INTEGER COUNT
C
      COMMON /RJ/ A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
      COMMON /RJ/ ACR,BLFEC
      COMMON /RJ/ CDASUB,CDASUF,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
      COMMON /RJ/ ER,ERRL,ERLL,ETAC,ETAN,F
      COMMON /RJ/ FAR,G,GAMMA,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA6
      COMMON /RJ/ H,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
      COMMON /RJ/ COUNT,ICONV,ICONVP,IFIRST
      COMMON /RJ/ M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
      COMMON /RJ/ MACR,MACRS,PM
      COMMON /RJ/ P,P1,P2,P3,P4,P5,P6,PT3OI,PRAR
      COMMON /RJ/ PT,PT1,PT2,PT3,PT4,PT5,PT6,PMOPT,PMOPT1,PMOPT2,PMOPT3
      COMMON /RJ/ PHI1,PHI2,PHI3,PHI4,PHI5,PHI6,PHIPRI,Q
      COMMON /RJ/ RADEG,R,SA,SART,SQRTOT
      COMMON /RJ/ T,T1,T2,T3,T4,T5,T6,TIME
      COMMON /RJ/ TT,TT1,TT2,TT3,TT4,TT5,TT6
      COMMON /RJ/ V,V1,V2,V3,V4,V5,V6
      COMMON /RJ/ WA,WF,X,Y,Z
C
C
      DATA A11,B11,C11,D11,E11,F11,G11
     * /-1407.66, .95416, -8.59033, 43.7266, 1468.91, 80484.,
     * -27706.6./
C
      FAR2=FAR*FAR
      FOVER=1./(1.+FAR)
```

```
      TERM1=B11+C11*FAR+D11*FAR2
      TERM2=E11+F11*FAR+G11*FAR2
      TTR=1.E*TT
      TT4I=A11+TTR*TERM1+FCVER*TERM2
      STT4I=SQRT(TT4I)
      ETAC=40.246*STT4I-0.3513*TT4I-1055.08
      ETAC=ETAC/100.
      IF(ER.LT. .0001) ETAC=0.0
      RETURN
      END
C
C
C
      SUBROUTINE HCRA
C
      DIMENSION PS(66), PSTAT1(11), FUELF(11)
C
C
C
      NWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
C     CPIA NOMENCLATURE
C
      REAL ISP,ISTAR,M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
      REAL MACR,MACRS
      INTEGER CCUNT
C
      COMMON /RJ/ A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
      COMMON /RJ/ ACR,BLEED
      COMMON /RJ/ CDASUB,CDASUF,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
      COMMON /RJ/ ER,ERFL,ERLL,ETAC,ETAN,F
      COMMON /RJ/ FAR,G,GAMMA,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA6
      COMMON /RJ/ H,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
      COMMON /RJ/ CCUNT,ICCNV,ICCNVP,IFIRST
      COMMON /RJ/ M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
      COMMON /RJ/ MACR,MACRS,PM
      COMMON /RJ/ P,P1,P2,P3,P4,P5,P6,PT3DI,PRAR
      COMMON /RJ/ PT,PT1,PT2,PT3,PT4,PT5,PT6,PMOPT,PMOPT1,PMOPT2,PMOPT3
      COMMON /RJ/ PHI1,PHI2,PHI3,PHI4,PHI5,PHI6,PHIPRI,Q
      COMMON /RJ/ RADEG,R,SA,SART,SQRTOT
      COMMON /RJ/ T,T1,T2,T3,T4,T5,T6,TIME
      COMMON /RJ/ TT,TT1,TT2,TT3,TT4,TT5,TT6
      COMMON /RJ/ V,V1,V2,V3,V4,V5,V6
      COMMON /RJ/ WA,WF,X,Y,Z
C
C
C     ANGLE OF ATTACK   PS(I) I=3,14
C     MACH NUMBER       PS(I) I=15,81
C
      DATA PS/
```

```
     A12.0,4.0,0.0,0.5,1.0,1.5,2.0,
     A2.5,3.0,4.0,5.0,6.0,7.0,8.0,
     A1.5,2.0,2.5,3.0,
     A .98407,.93504,.89727,.88386,
     A .98069,.93504,.89727,.88386,
     A .97828,.93504,.89689,.88275,
     A .97586,.93410,.89463,.88165,
     A .97345,.93295,.89313,.87833,
     A .97056,.93039,.89087,.87501,
     A .96815,.92808,.88711,.87058,
     A .96320,.92158,.87733,.85842,
     A .95704,.91044,.86076,.84183,
     A .94068,.89257,.83781,.81749,
     A .94111,.87331,.81185,.78210,
     A .92505,.85267,.79190,.75666/
C
      DATA PSTAT1/3.07,3.9,4.9,6.02,7.,8.,9.1,10.,11.,12.,13.18/
C
      DATA FUELF/.94,1.130,1.375,1.628,1.859,
     A 2.056,2.250,2.416,2.562,2.722,3.05/
C
      IF (IFIRST) 10,20,20
   10 CONTINUE
      WRITE (6,50)
      RETURN
   20 CONTINUE
      IF (CCUNT-1) 30,30,40
   30 CONTINUE
      CALL INTR2C (ALPHA,M,PS1CP,PS,1)
      PS1=PS1CP*(P/6894.757)
      TFFPS=CDLI(PS1,PSTAT1,FUELF,11)
      TFF=TFFPS*.4535924
      WF=TFF
   40 RETURN
C
   50 FORMAT (T6,'FUEL CONTROL FOR STV G      ')
C
      END
C
C
C
      SUBROUTINE HCF
C
      DIMENSION PR(81), AA(81), CADD(12), XM(12), BWR(12)
C
C     NWC COMMON DECK FOR AIRBREATHING PROPULSION PROGRAMS
C     CPIA NOMENCLATURE
C
```

```
      REAL ISP,ISTAR,M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
      REAL MACR,MACRS
      INTEGER COUNT
C
      COMMON /RJ/ A,AINF,A1,A2,A3,A4,A5,A6,AR,AFR,AFS,ALPHA
      COMMON /RJ/ ACR,BLEED
      COMMON /RJ/ CDASUB,CDASUP,CDB,CF,CFINF,CF6,CFB,CFT,CFC,CNM
      COMMON /RJ/ ER,ERRL,ERLL,ETAC,ETAN,F
      COMMON /RJ/ FAR,G,GAMMA,GAMMA1,GAMMA2,GAMMA3,GAMMA4,GAMMA5,GAMMA6
      COMMON /RJ/ H,ISP,ISTAR,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
      COMMON /RJ/ COUNT,ICONV,ICONVP,IFIRST
      COMMON /RJ/ M,M1,M2,M3,M4,M5,M6,MW,MW1,MW2,MW3,MW4,MW5,MW6
      COMMON /RJ/ MACR,MACRS,PM
      COMMON /RJ/ P,P1,P2,P3,P4,P5,P6,PT30I,PRAR
      COMMON /RJ/ PT,PT1,PT2,PT3,PT4,PT5,PT6,PMOPT,PMOPT1,PMOPT2,PMOPT3
      COMMON /RJ/ PHI1,PHI2,PHI3,PHI4,PHI5,PHI6,PHIPRI,Q
      COMMON /RJ/ RADEG,R,SA,SART,SQRTOT
      COMMON /RJ/ T,T1,T2,T3,T4,T5,T6,TIME
      COMMON /RJ/ TT,TT1,TT2,TT3,TT4,TT5,TT6
      COMMON /RJ/ V,V1,V2,V3,V4,V5,V6
      COMMON /RJ/ WA,WF,X,Y,Z
C
C
C     PRESSURE RECOVERY INSTALLED ALVJ -- BETA=0.0  (PS(I) I=19,81)
C
      DATA PR/
     A5.0,7.0,0.0,0.9,1.0,1.5,2.2,
     A2.43,2.7,3.1,4.0,0.0,1.0,2.0,
     A4.0,6.0,8.0,10.0,
     A .9500,.9475,.9450,.9400,.9250,.8775,.8320,
     A .9500,.9475,.9450,.9400,.9250,.8775,.8320,
     A .9500,.9475,.9450,.9400,.9250,.8775,.8320,
     A .9400,.9375,.9350,.9300,.9100,.8600,.8150,
     A .7675,.7660,.7640,.7600,.7430,.7000,.6575,
     A .6980,.6860,.6840,.6800,.6350,.5750,.5275,
     A .5400,.5375,.5350,.5300,.4925,.4300,.3800,
     A .3800,.3760,.3725,.3650,.3400,.2850,.2400,
     A .2000,.1960,.1925,.1850,.1600,.1050,.0800/
C
C
C     AIR CAPTURE INSTALLED ALVRJ DEC 1975   AA(I) I=19,81
C
      DATA AA/
     A5.0,7.0,0.0,0.9,1.0,1.5,2.2,
     A2.43,2.7,3.1,4.0,0.0,1.0,2.0,
     A4.0,6.0,8.0,10.0,
     A .9465,.9457,.9452,.9440,.9400,.9350,.9270,
     A .9465,.9457,.9452,.9440,.9400,.9350,.9270,
```

```
     A .5825,.5819,.5812,.5800,.5700,.5650,.5050,
     A .6725,.6712,.6700,.6675,.6520,.6400,.6275,
     A .9010,.9000,.8990,.8970,.8725,.8550,.8350,
     A .9425,.9400,.9375,.9325,.9200,.9075,.8950,
     A .9450,.9425,.9410,.9375,.9300,.9225,.9125,
     A .9450,.9437,.9425,.9400,.9370,.9325,.9250,
     A .9450,.9437,.9425,.9400,.9370,.9325,.9250/
C
C
      DATA XM/
     A  .5,.9,1.,1.4,1.6,2.2,2.43,2.48,2.5,2.7,3.0,3.25/
C
      DATA CADC/
     A .0213,.0213,.0132,.125,.0988,.0221,.0184,.0173,.0171
     B ,.0123,.0149,.0176/
C
      DATA BWR/
     A .0339,.0339,.0208,.0237,.0249,.0334,.0287,.0265,.0259,
     B .0213,.0171,.0150/
C
      IF (IFIRST) 10,20,20
   10 CONTINUE
      PRAR=0.0
      TOL=.0004
      WRITE (6,100) TOL
      RETURN
C
C
   20 CONTINUE
C
C SET PRESENT PR/AR EQUAL TO PREVIOUS PR/AR
C
      PRARP=PRAR
C
C CALCULATE NEW PR/AR
C
      TT=T/FTOTT(1.4,M)
      TTRATO=SQRT(TT3/TT)
      PRAR=TTRATO*PMOPT*MACRS*A1/(PMOPT3*A3)
      IF (IBC.EQ.0) PRAR=TTRATO*PMOPT*MACRS*A1*(1.0+FAR)/(PMOPT3*A3)
C
C CHECK FOR CONVERGENCE BY COMPARING PR/AR TO PREVIOUS PR/AR
C
      IF (ABS(PRARP-PRAR)-TOL) 40,40,30
   30 ICONV=1
   40 CONTINUE
      CALPHA=COS(ALPHA/RADEG)
C
```

```
C
C     COMPUTE CRITICAL PRESSURE RECOVERY AND MAXIMUM AIR CAPTURE RATIO
C
      ALPHAB=ABS(ALPHA)
      CALL INTF20 (M,ALPHAB,MACR,AA,1)
      BLDTS=CDL1(M,XM,BWR,12)
      BLDDB=.015
      BLEED=BLDTS+BLDDB
      MACRS=MACR*(1.0-BLEED)
      CALL INTF20 (M,ALPHAB,PRIAD,PR,1)
C
C     CRITICAL PRESSURE RECOVERY
C
      PACR=PRIAD
      IF (PRAR-PACR) 50,60,70
C
C     PRESSURE RECOVERY  .LT. CRITICAL
C
   50 PT3OI=PRAR
      ACR=1.
      GO TO 80
C
C     PRESSURE RECOVERY  .EQ. CRITICAL
C
   60 PT3OI=PRIAD
      ACR=1.
      GO TO 80
C
C     PRESSURE RECOVERY  .GT. CRITICAL --CHANGE AIR CAPTURE AND SET PRES
C     RECOVERY EQUAL TO CRITICAL VALUE
C
   70 PT3OI=PRIAD
      ACR=PRIAD/PRAR
   80 POPT=PPCPT(1.4,M)
      PT=P/PCPT
      PT3=PT*PT3OI
      FM=(PRIAD-PT3OI)/(PRIAD/100.)
      AINF=ACR*A1*MACRS
      SQTINF=SQRT(TT)
      WA=(AINF*PT*FMOPT)/SQTINF
      IF(ICONVF.NE.0 .OR. ICONV.EQ.1) RETURN
      CDSUB=0.0
      IF (PRAR.LE.PACR) GO TO 90
C
C     CALCULATE ADDITIVE DRAG DUE TO SUBSONIC SPILLAGE
      ETASUB=M/((M*10.2095)-11.5652)
      CDSUB=2.*(1.-ETASUB)*(1.-ACR)
C
```

```
C      CALCULATE ADDITIVE DRAG DUE TO SUPERSONIC SPILLAGE
C
   90 CAED=CDLI(M,XM,CADD,12)
      IF (PRAR.LE.PACR) CDSUB=C.0
      CDASUB=CDSUB*A1/AR
      CDASUP=CAED
      CFB=C.C
C                           CFT IS RAT DRAG
      CFT=C.CC32
       PESI=2.14678-1.11439*M+.22989*M*M
      PE=PESI*F
      AINFB=A1*BLDCB
      AE=9.33*.00064516
      TRM1=P*AINFB*1.4*M*M
      TRM2=P*AE
      WDCT=(FT*AINFB*PMOPT)/SQTINF
      XMCE=WDCT*SQRT(TT-222.)/(PE*AE)
      XME=FRCM(1.4,XMCE)
      TRM3=PE*AE*(1.+1.4*XME*XME)
      CFC=-((TRM1+TRM2-TRM3)/(C*AR))
      RETURN
C
 100  FORMAT (T6,'ALVRJ INLET DATA DEC 1975 ,TOL=',F7.6)
C
      END
C
C
C
      FUNCTION INTRP (AH,AL,BH,BL,FH,FM)
C
      REAL INTRP
      C2 = AL + (AH - AL)*FM
      C1 = BL + (BH - BL)*FM
      INTRP = C1 + (C2 - C1)*FH
      RETURN
      END
C
C
C
      FUNCTION FATAS(GAM,XM)                                                    FATA0030
C
CFATAS    FUNCTION SUBPROGRAM TO CALCULATE A OVER ASTAR                         FATA0010
C           FROM GAMMA AND MACH NUMBER                                          FATA0020
C
      Z=(GAM+1.C)/(2.0*(GAM-1.C))                                               FATA0040
      A=2.C/(GAM+1.0)                                                           FATA0050
      B=(GAM-1.0)/2.0                                                           FATA0060
      C=1.C/XM                                                                  FATA0070
```

```
      D=A*(1.0+(B*XM*XM))                                               FATA0080
      FATAS=C*(D**Z)                                                    FATA0090
      RETURN                                                            FATA0100
      END                                                               FATA0110
C
C
C
      FUNCTION FIXAC (A)                                                R    2
C
      TFIX=ABS(A)-1.0000005                                             R    4
      IF (TFIX.LT.0.0) GO TO 10                                         R    6
   10 IF (A.LT.0.0) GO TO 20                                            R    8
      FIXAC=0.0                                                         R   10
      GO TO 30                                                          R   12
   20 FIXAC=3.14159265                                                  R   14
   30 CONTINUE                                                          R   16
      RETURN                                                            R   18
      END                                                               R   20-
C
C
C
      FUNCTION FIXAS (A)                                                S    2
C
      TFIX=ABS(A)-1.0000005                                             S    4
      IF (TFIX.LT.0.0) GO TO 10                                         S    6
   10 IF (A.LT.0.0) GO TO 20                                            S    8
      FIXAS=1.57079633                                                  S   10
      GO TO 30                                                          S   12
   20 FIXAS=-1.57079633                                                 S   14
   30 CONTINUE                                                          S   16
      RETURN                                                            S   18
      END                                                               S   20-
C
C
C
      FUNCTION  FMARH (GAM,ATAS)                                        FMAR0030
C
CFMARH          FUNCTION SUBPROGRAM TO CALCULATE MACH NUMBER            FMAR0010
C                            FROM ATAS AND GAMMA                        FMAR0020
C
      Z = (GAM + 1.0) / (2.0*(GAM - 1.0))                               FMAR0040
      DXMG = 10.0                                                       FMAR0050
      XMG = 10.0                                                        FMAR0060
  500 DXMG = DXMG / 2.0                                                 FMAR0070
      XMG= AMAX1(1.0,AMIN1(100.0,XMG))                                  FMAR0080
      IF(DXMG - .0000015)100,100,200                                    FMAR0090
  200 CATAS = 1.0/XMG*((2.0+ (GAM - 1.0) *XMG **2 )/(GAM + 1.0))**Z     FMAR0100
      IF(CATAS - ATAS)700,100,800                                       FMAR0110
```

```
  700 XMG = XMG + DXMG                                          FMAR0120
      GO TO 500                                                  FMAR0130
  800 XMG = XMG - DXMG                                          FMAR0140
      GO TO 500                                                  FMAR0150
  100 FMARH = XMG                                               FMAR0160
      RETURN                                                     FMAR0170
      END                                                        FMAR0180
C
C
C
      FUNCTION FMARL(GAM,AR)
C
C      FUNCTION ROUTINE TO FIND MACH FROM A/A* AND GAMMA
C      USING NEWTONS METHOD FOR FINDING ROOTS OF AN EQUATION
C
C      SUBSONIC SOLUTION
      IF(GAM.LT.1.) WRITE(6,30)
      IF(AR.LT.1.) WRITE(6,40)
      IF(GAM.LT.1. .OR. AR.LT.1.) CALL EXIT
      GM1=GAM-1.
      GP1=GAM+1.
      A=2./GP1
      B=GM1/GP1
      Z=GP1/(2.*GM1)
      IF(AR.LT.1.3401) XMP=2.4706-(1.47059*AR)
      IF(AR.GT.1.34 .AND. AR.LT.3.001) XMP=.77443-(.2048*AR)
      IF(AR.GT.3.) XMP=(A**Z)/AR
      TOL=.0001
   10 CONTINUE
      BRACK=A+(B*XMP*XMP)
      POWER=BRACK**Z
      TOP=(1./XMP)-(AR/POWER)
      BOTTOM=(1./BRACK)-(1./(XMP*XMP))
      XM=XMP-(TOP/BOTTOM)
      DELTA=ABS(XM-XMP)
      XMP=XM
      IF(DELTA.GT.TOL) GO TO 10
C
      FMARL=XM
C
      RETURN
C
   30 FORMAT('GAMMA LT 1 IN FMARL')
   40 FORMAT('A/A* LT 1 IN FMARL')
C
      END
C
C
```

```
C
      FUNCTION  FMPHIM  (GAM , PHI )                                    FMPH0020
C
CFMPHIM    FUNCTION SUBPROGRAM TO CALC MACH NO  FROM PHIM                FMPH0010
C
      TOL = .00001                                                      FMPH0030
      XMSN = 1.0 / ( PHI**3 )                                           FMPH0040
  127 X = XMSN                                                          FMPH0050
      XMSN = (1.0 + (GAM*(XMSN**2))) / (PHI * SQRT(2.0 * (GAM+1.0)*(1.0 FMPH0060
     1+(GAM-1.0)*.5*(XMSN**2))))                                        FMPH0070
      XMSN= AMAX1(0., AMIN1(1.0,XMSN))
      IF (ABS(X-XMSN)-TOL ) 125, 125, 128                               FMPH0090
  128 XMSN = (X + XMSN) / 2.0                                           FMPH0100
      GO TO 127                                                         FMPH0110
  125 FMPHIM = XMSN                                                     FMPH0120
      RETURN                                                            FMPH0130
      END                                                               FMPH0140
C
C
C
      FUNCTION  FMPOPT (GAM,POPT)                                       FMP00030
C
CFMPOPT   FUNCTION  SUBPROGRAM TO CALC  XMACH                           FMP00010
C                   FROM  P/PT AND GAMMA                                FMP00020
C
      Z =  (POPT) ** ((1.0- GAM)/ GAM)                                  FMP00040
      SMPPT =   ( Z - 1.0)/(GAM - 1.0) * 2.0                            FMP00050
      FMPOPT = SQRT(SMPPT)                                              FMP00060
      RETURN                                                            FMP00070
      END                                                               FMP00080
C
C
C
      FUNCTION PHIM (GAM, XMCH)                                          PHI0020
C
C PHIM  FUNCTION  TO CALC PHI FROM GAMMA AND MACH                        PHI0010
C
      PHIM = (1.0+GAM *(XMCH **2)) /(XMCH  * SQRT(2.0*(GAM + 1.0)*(1.0+  PHI0030
     1 .5 *(GAM -1.0) * (XMCH **2))))                                    PHI0040
      RETURN                                                             PHI0050
      END                                                                PHI0060
C
C
C
      FUNCTION  FPOPT (GAM,XMACH)                                        FPO0030
C
C FPOPT    FUNCTION  SUBPROGRAM TO CALC TOTAL PRESSURE RATIO             FPO0010
C              FROM GAMMA AND MACH NO.                                   FPO0020
```

```
C
      Z = -GAM /(GAM - 1.0)                                              FP00040
      FPOPT  =      (1.0+.5*(GAM -1.0)*(XMACH**2))**Z                    FP00050
      RETURN                                                             FP00060
      END                                                                FP00070
C
C
C
      FUNCTION FRCM(GAM,XMCIR)
C
C     FUNCTION ROUTINE TO COMPUTE MACH NUMBER FROM
C     MCIRCLE AND GAMMA FOR ALL MACH NUMBERS USING NEWTONS METHOD
C
      IF (GAM.LT.1.) WRITE (6,100)
      IF (GAM.LT.1.) CALL EXIT
      A=GAM
      B=0.5*GAM*(GAM-1.)
      C=XMCIR/.C5902063
      XMP=C/(1.18+(.0923*C))
      TOL=.0001
   10 CONTINUE
      BRACK=A+(B*XMP*XMP)
      POWER=SQRT(BRACK)
      TOP=(XMP*POWER)-C
      BOTTOM=(A+(2.*B*XMP*XMP))/POWER
      XM=XMP-(TOP/BOTTOM)
      DELTA=ABS(XM-XMP)
      XMP=XM
C
      IF(DELTA.GT.TOL) GO TO 10
C
      FRCM=XM
C
      RETURN
C
  100 FORMAT('GAMMA LT 1 IN FROM')
      END
C
C
C
      FUNCTION FTCTT  (GAM,XMACH)                                        FT00020
C
C FTCTT   FUNCTION SUBPROGRAM TO CALC  T/TT    FROM    GAMMA AND MACH NC FTC0010
C
      FTOTT   =  (1.0+.5*( GAM-1.0)*(XMACH**2))**(-1.0)                  FT00030
      RETURN                                                             FT00040
      END                                                                FT00050
C
```

```
C
C
      SUBROUTINE TAB(FAS)
C
      WRITE(6,15)
   15 FORMAT (T6,'NO.2 JP5 AND JETA FUEL MAY 1,1976 ONE ATM          ')
      FAS=    14.6354
      RETURN
      END
C
C
C
      FUNCTION FMCIR (GAM,XM)
C
C MCIRCLE      FUNCTION SUBPROGRAM TO CALC MCIR FROM GAMMA AND MACH NO
C        R AIR= 8314.34/28.96246 = 287.073 N M/ KG K
C      M CIRCLE DIMENSIONS = SEC * SQRT(K) / METRE
C
      FMCIR=.0590206289*XM*SQRT(GAM*(1.0+((GAM-1.0)/2.0)*XM**2))
      RETURN
      END
C
C
C
      SUBROUTINE AIR (Z,V,J,P,T,S,D,Q,GO,K,KK)
C
C     Z=ALTITUDE IN METERS
C     V=MACH NUMBER OR VELOCITY, SEE KK
C     J=FIRST PASS IDENTIFIER SWITCH, -=FIRST PASS AND +=NOT FIRST PASS
C     P=PRESSURE IN PASCALS OR PSI
C     T=TEMPERATURE IN K OR R
C     S=SPEED OF SOUND IN METERS/SEC OR FT/SEC
C     D=DENSITY IN KG/CUBIC METER OR LBM/CUBIC FOOT
C     Q=DYNAMIC PRESSURE IN PA OR PSI
C     GO=LOCAL GRAVITY IN FT/SEC2 OR METERS/SEC2
C     K=1, SI UNITS
C     K=2, ENGLISH UNITS
C     KK=1, V IS MACH NUMBER
C     KK=2, V IS VELOCITY, METERS/SEC OR FT/SEC
C
      DIMENSION A(2,16), C(15)
C
C **                     A(1,J),J=1,16
C **  THESE ARE THE CHEBYSHEV COEFFICIENTS FOR P/PO
C
C **                     A(2,J),J=1,16
C **  THESE ARE THE CHEBYSHEV COEFFICIENTS FOR D/DO
C
```

```
      DATA A/
     A-4.4038210, -4.0135170, -2.2874890, -2.1765372,
     B -.0494300, -.1358554,   .0355105,   .0498002,
     C -.0055256,   .0061961, -.0030910, -.0121108,
     D  .0026116,   .0029277, -.0004221,   .0044266,
     E -.0010261, -.0039804,   .0007523, -.0013126,
     F  .0003149,   .0033573, -.0005513, -.001147,
     G -.0000031, -.0021286,   .0002918,   .0008523,
     H -.0001420,   .0010109,   .99998983, 1.0000000/
C
C **   SET CONSTANTS
C
      DATA PC,DC,RBAR,GC,Z1/
     *   101325.0,1.225,287.07299,9.80665,30480.0/
C
C **   IF NECESSARY , CONVERT TO SI UNITS
C                           Z BECOMES METERS
C                           V BECOMES METERS/SEC
C
      IF (K.EQ.2) Z=Z*.3048
      IF (K.EQ.2.AND.KK.EQ.2) V=V*.3048
C
C **   START HERE
C
      IF (J) 10,20,20
   10 WRITE (6,60)
      WRITE (6,70) Z1
C
C **   COMPUTE C(K) COEFFICIENTS
C
   20 ETA=2.*(2.*Z/Z1-1.)
      C(1)=ETA
      C(2)=ETA**2-2.
      DO 30 I=3,15
   30   C(I)=ETA*C(I-1)-C(I-2)
C
C **   COMPUTE CHEBYSHEV TRUNCATED EXPANSION
C
      ALNP=A(1,1)
      ALND=A(2,1)
      DO 40 I=2,15
        ALNP=ALNP+A(1,I)*C(I-1)
   40   ALND=ALND+A(2,I)*C(I-1)
C
C **   COMPUTE ATMOSPHERIC PRESSURE
C                           P IN N/SQ METER
C
      ALNP=.5*ALNP
```

```
      FOFZ=EXF(ALNP)
      F = FCFZ*FC*A(1,16)
C
C **  CCMPUTE GRAVITY AT ALTITUDE AND LATITUDE
C                         RC IS EQUATORIAL RADIUS (M) PER CRC HANDBOOK
C                         GC IN M/SEC SQ
C
      RC=6376377.45
      GO=GC*((RC/(RO+Z))**2)
C
C **  CCMPUTE ATMCSPHERIC DENSITY
C                         D IN KG PER CUBIC METRE
C
      ALND=.5*ALNC
      DODZ=EXF(ALND)
      D = DCDZ*DC*A(2,16)
C
C **  COMPUTE AIR TEMPERATURE
C                         T IN DEGREE KELVIN
C
      T=P/(REAR*D)
C
C **  COMPUTE SPEED OF SCUND IN M/SEC
C
      S=SQRT(401.90219*T)
C
C **  COMPUTE DYNAMIC PRESSURE IN PA
C
      IF (KK.EC.1) Q=0.7*F*V**2
      IF (KK.EC.2) Q=0.7*P*(V/S)**2
C
C **  CONVERT TC ENGLISH UNITS IF NECESSARY
C
      IF (M.EC.1) GC TC 50
      Z=Z/.3048
      IF(KK.EC.2) V=V/.3048
      P=P/6894.757
      T=T*1.8
      S=S/.3048
      D=D/16.01846
      C=C/6894.757
      GC=GC/.3048
   50 RETURN
   60 FORMAT (T9,'*** US 1962 STANDARD ATMOSPHERE')
   70 FORMAT (T9,'*** ALTITUDE FROM 0 TO ',F8.0,' METERS')
      END
C
C
```

```
C
      FUNCTION ANORM(A,B)
C
C     CALCULATES TOTAL PRESSURE RATIO ACROSS A NORMAL SHOCK
C     A=GAMMA,B=XMACH
C
      C=(A+1.)/2.
      D=(A-1.)/2.
      E=(2.*A)/(A+1.)
      F=(A-1.)/(A+1.)
      G=A/(A-1.)
      H=1./(A-1.)
      BB=B*B
      ANUM=(((C*BB)/(1.+D*BB))**G)
      DEM=((E*BB-F)**H)
      ANORM = ANUM/DEM
      RETURN
      END
C
C
C
      FUNCTION FPMOPT( GAM , XM )
C      FUNCTION SUBPROGRAM TO CALCULATE P/PT * M CIRCLE
C      FROM GAMMA AND MACH NUMBER
C      DIMENSIONS = SEC * SQRT(K) / METRE
C
      FPMOPT = FMCIR ( GAM , XM ) * FPOPT ( GAM , XM )
      RETURN
      END
C
C
C
      SUBROUTINE SERCH (X,A,J,I,M)
C
      DIMENSION A(20)
      DO 10 L=1,I
      J=L
      IF (X.LE.A(L)) GO TO 20
   10 CONTINUE
   20 IF (J.EQ.1) GO TO 30
      J=J-1
   30 RETURN
      END
C
C
C
      FUNCTION SHOCK(XM,D)
C
```

```
C SHOCK FUNCTION SUBPROGRAM TO CALCULATE THE OBLIQUE SHOCK ANGLE FROM
C       THE WEDGE ANGLE AND MACH NUMBER
C         XM=MACH NUMBER  D=WEDGE ANGLE DELTA IN RADIANS
C       SHOCK=SHOCK ANGLE THETA IN RADIANS
C       P,Q,R,A,B=CONSTANTS  F=PHI WHICH IS AN INTERMEDIATE ANGLE
C         RADEG=TRANSFORMS RADIANS TO DEGREES
C
      RADEG = 57.2957795131
      SINZDZ=SIN(D)
      P=     -((XM*XM+2.)/(XM*XM)) -(1.4*SINZDZ*SINZDZ)
      OQ=1.44+(.4/(XM*XM))
      Q=((2.*XM*XM)+1.)/(XM*XM*XM*XM)+(OQ*SINZDZ*SINZDZ)
      C=COS(D)
      R=(-C*C)/(XM*XM*XM*XM)
      A=(1./3.)*(3.*Q-P*P)
      B=(1./27.)*(2.*P*P*P-9.*P*Q +27.*R)
      TEST = ((B*B)/4.)+ ((A*A*A)/27.)
      IF(TEST)1,1,2
    1 CONTINUE
      F = ARCOS((-B/2.)/SQRT((-1./27.)*A*A*A))
      FO=F*RADEG
      FOX=(FO/3.)+240.
      FOXE=FOX/RADEG
      X=2.*SQRT(-A/3.)*COS(FOXE)
      SHOCK = ARSIN(SQRT(X-P/3.))
      RETURN
    2 WRITE(6,3)
    3 FORMAT(48H HOW COME THE INPUTS TO SHOCK GIVE COMPLEX ROOTS)
      RETURN
      END
C
C
C
      FUNCTION STDLI (TBX,TBY,TAB,X,Y,NX,NY)
C
C                FUNCTION DESIGNED TO ALLOW THE USER TO INPUT
C        VARIABLE LENGTH TABLES.  IT INTERPOLATES FIRST IN THE X
C        VARIABLE AND THEN IN THE Y VARIABLE.  IT EXTRAPOLATES FROM
C        THE FIRST OR LAST INTERVAL IF THE X OR Y IS OUTSIDE OF THE
C        BOUNDS OF THE TABLE.  IT REQUIRES THE SUBROUTINE SERCH FOR
C        ITS OPERATION.
C
C             PROGRAMMER -- H.SOBEL    30 OCTOBER 1967
C
      DIMENSION TBX(NX), TBY(NY), TAB(11,11)
      CALL SERCH (X,TBX,I,NX,1)
      CALL SERCH (Y,TBY,J,NY,1)
      XX=TBX(I)
```

```
      YY=TBY(J)
      TXY=TAB(I,J)
      XJ=TAB(I,J)+(X-TBX(I))*((TAB(I+1,J)-TAB(I,J))/(TBX(I+1)-TBX(I)))
      XJ1=TAB(I,J+1)+(X-TBX(I))*((TAB(I+1,J+1)-TAB(I,J+1))/(TBX(I+1)-TBX
     1(I)))
      STOL=XJ+(Y-TBY(J))*((XJ1-XJ)/(TBY(J+1)-TBY(J)))
      STOL1=STOL
   10 RETURN
      END
C
C
C
      FUNCTION STOLIA (TBX,TBY,TAB,X,Y,NX,NY)
C
C                FUNCTION DESIGNED TO ALLOW THE USER TO INPUT
C        VARIABLE LENGTH TABLES.  IT INTERPOLATES FIRST IN THE X
C        VARIABLE AND THEN IN THE Y VARIABLE.  IT EXTRAPOLATES FROM
C        THE FIRST OR LAST INTERVAL IF THE X OR Y IS OUTSIDE OF THE
C        BOUNDS OF THE TABLE.  IT REQUIRES THE SUBROUTINE SERCH FOR
C        ITS OPERATION.
C
C             PROGRAMMER -- H.SOBEL    30 OCTOBER 1967
C
      DIMENSION TBX(NX), TBY(NY), TAB(NX,NY)
      CALL SERCH (X,TBX,I,NX,IC)
      CALL SERCH (Y,TBY,J,NY,JC)
      XJ=TAB(I,J)+(X-TBX(I))*((TAB(I+1,J)-TAB(I,J))/(TBX(I+1)-TBX(I)))
      XJ1=TAB(I,J+1)+(X-TBX(I))*((TAB(I+1,J+1)-TAB(I,J+1))/(TBX(I+1)-TBX
     1(I)))
      STOL1=XJ+(Y-TBY(J))*((XJ1-XJ)/(TBY(J+1)-TBY(J)))
      STOLIA=STOL1
   10 RETURN
      END
C
C
C
      SUBROUTINE SUBSON(H,ZM,F,C,W,Q,A)
C
      C=.3
      SFC=3.0
      THR=.224809*C*Q*A
      WF=(SFC*THR)/3600.
      W=WF*.45359
      IF(W.LT.  0.0)W=-W
      RETURN
      END
C
C
```

```
C
      FUNCTION VECMAG(X)                                                VECM001C
C
      DIMENSION X(3)                                                    B    2
   10 X2=X(1)*X(1)+X(2)*X(2)+X(3)*X(3)                                  B    4
      VECMAG=SQRT(X2)                                                   B    6
      RETURN                                                            B    8
      END                                                               B   10-
C
C
C
      FUNCTION TC (T,ER,ATM)
C
C     THEORETICAL TEMPERATURE RISE CORRECTION FOR PRESSURES OTHER
C     THAN ONE ATMOSPHERE (HYDROCARBON FUELS)
C
      DIMENSION C(8), E(31), G(31,3), P(13), H(13,2)
      DATA C /-.129403E4,-.191763E3,.746699E2,-.434459E2,-.397487E2,-.26
     17021E2,.13558305E2,-.10890892E2/
      DATA E /.0,.05,.1,.15,.2,.25,.3,.35,.4,.45,.5,.55,.6,.65,.7,.75,.8
     1,.85,.9,.95,1.,1.05,1.1,1.15,1.2,1.25,1.3,1.35,1.4,1.45,1.5/
      DATA G /0.,.000052,-.00007,-.000258,-.00084,-.001837,-.00407,-.007
     1901,-.01473,-.025548,-.04194,-.065907,-.09719,-.136621,-.18292,-.2
     236296,-.29312,-.351881,-.40644,-.453286,-.48303,-.481847,-.45954,-
     3.418049,-.36713,-.316900,-.26822,-.225566,-.18821,-.157010,-.13168
     4,0.,.000144,-.00045,-.001341,-.00385,-.008259,-.01724,-.032921,-.0
     55722,-.093804,-.13956,-.198405,-.25292,-.291575,-.30062,-.263938,-
     6.17710,-.014879,.17645,.395969,.53284,.429298,.22521,-.007856,-.22
     7277,-.332987,-.38518,-.391398,-.36897,-.341909,-.30221,0.,.000244,
     8.00173,.003242,.00843,.019739,.03957,.071769,.11444,.173894,.22936
     9,.270517,.27377,.201127,.08546,-.084450,-.21432,-.205284,-.08215,.
     A284744,.53085,.162602,-.30329,-.468888,-.47144,-.264349,-.00121,.1
     B63745,.27903,.341386,.351907
      DATA P /.20,.35,.50,.75,1.0,1.5,2.0,3.5,5.0,7.5,10.,15.,20./
      DATA H /.44161,.297035,.18097,.066615,0.,-.102718,-.16656,-.298846
     1,-.36320,-.446764,-.49337,-.574833,-.60759,-.80970,-.477316,-.2282
     27,-.048428,0.,.075181,.08051,.061939,-.02264,-.121555,-.21910,-.37
     33927,-.48712/
C
C     CHECK FOR ENTRY OUTSIDE LATTICE
C
      IF (T.LT.298.) GO TO 50
      IF (T.GT.1300.) GO TO 60
      IF (ER.GT.1.5) GO TO 70
      IF (ATM.LT..2) GO TO 90
      IF (ATM.GT.20.) GO TO 80
C
C     COMPUTE FACTOR ANALYSIS FACTORS F1,F2,F3
```

```
C
      F1=-((.227276E-3+.100807E-6*T)*T+.018785)
      F2=(.00012C097-.1254E-5*T)*T+.0846043
      F3=(((.113312E-7-.244175E-11*T)*T-.141615E-4)*T+.00496295)*T+.0056
     A0473
C
C   LINEARLY INTERPOLATE FOR FACTORS G1,G2,G3
C
      DO 10 I=1,31
      IF (ER.LE.E(I)) GO TO 20
   10 CONTINUE
   20 J=I-1
      R=(ER-E(J))/(E(I)-E(J))
      G1=G(J,1)+(G(I,1)-G(J,1))*R
      G2=G(J,2)+(G(I,2)-G(J,2))*R
      G3=G(J,3)+(G(I,3)-G(J,3))*R
C
C   LINEARLY INTERPOLATE FOR FACTORS H1,H2
C
      DO 30 I=1,13
      IF (ATM.LE.P(I)) GO TO 40
   30 CONTINUE
   40 J=I-1
      R=(ATM-P(J))/(P(I)-P(J))
      H1=H(J,1)+(H(I,1)-H(J,1))*R
      H2=H(J,2)+(H(I,2)-H(J,2))*R
C
C   COMPLTE TEMPERATURE CORRECTION (TC)
C
      TC=C(1)*F1*G1*H1+C(2)*F2*G2*H1+C(3)*F1*G2*H2+C(4)*F2*G1*H2
     D         + C(5)*F3*G3*H1
     E         + C(6)*F2*G3*H2
     F         + C(7)*F3*G2*H2
     G         + C(8)*F1*G3*H2
C
      RETURN
   50 WRITE (6,110)
      GO TO 100
   60 WRITE (6,120)
      GO TO 100
   70 WRITE (6,130)
      GO TO 100
   80 WRITE (6,150)
      GO TO 100
   90 WRITE (6,140)
  100 CALL EXIT
      RETURN
C
```

```
110  FORMAT ('0 TOTAL TEMPERATURE LESS THAN 298 K. END RUN.')
120  FORMAT ('0  TOTAL TEMPERATURE GREATER THAN 1300 K. END RUN.')
130  FORMAT ('0 FUEL EQUIVALENCE RATIO EXCEEDS 1.5 END OF RUN.')
140  FORMAT ('0 ATMOSPHERIC PRESSURE LESS THAN 0.2, END OF RUN.')
150  FORMAT ('0 ATMOSPHERIC PRESSURE EXCEEDS 20, END OF RUN.')
     END
C
C
C
     FUNCTION ODLI (A,X,F,N)
C
     DIMENSION X(N), F(N)
     IF (N-2) 50,10,10
  10 IF (X(1).EQ.X(N)) GO TO 60
     IF (X(1).EQ.X(2).OR.X(N-1).EQ.X(N)) GO TO 70
     CALL SERCH (A,X,I,N,D)
     IF (A.NE.X(I+1)) GO TO 30
     IF (X(I+1)-X(I+2)) 20,40,20
  20 ODLI=F(I+1)
     RETURN
  30 ODLI=F(I)+(A-X(I))*(F(I+1)-F(I))/(X(I+1)-X(I))
     RETURN
  40 ODLI=0.5*(F(I+1)+F(I+2))
     RETURN
  50 WRITE (6,90)
     GO TO 80
  60 WRITE (6,100)
     GO TO 80
  70 WRITE (6,110)
  80 CALL EXIT
  90 FORMAT (1H1,10X,57HERROR IN SUBROUTINE ODLI, TABLE HAS LESS THAN T
    1WO ENTRIES)
 100 FORMAT (1H1,10X,60HERROR IN SUBROUTINE ODLI, INITIAL AND FINAL POI
    1NTS ARE EQUAL)
 110 FORMAT (1H1,10X,59HERROR IN SUBROUTINE ODLI, DISCONTINUITY AT ONE
    1END OF TABLE)
     END
C
C
C
     SUBROUTINE INTR20(X1,X2,C,D,L)
C
     DIMENSION C(1000)
     X=X1
     Y=X2
     N1 =C(1)
     N2=D(2)
     IX1=N1+2
```

```
      DO 10 II=4,IX1
      I=II
      IF(X.LT.C(I)) GO TO 11
   10 CONTINUE
   11 IO =I-2
      IX2= N1+4
      IX3=IX1+N2
      DO 20 JJ=IX2,IX3
      J=JJ
      IF(Y.LT.C(J)) GO TO 21
   20 CONTINUE
   21 JO=J-IX1
      NA=IX3+N2*((L-1)*N1+IO-2)+JO-1
      F1=(X-C(I-1))/(C(I)-C(I-1))
      F2=(Y-C(J-1))/(C(J)-C(J-1))
      F3=F1*F2
      C= C(NA)*(1.-F1-F2+F3)
     1  + C(NA+N2)*(F1-F3)
     2  + C(NA+1)*(F2-F3)
     3  + C(NA+N2+1)*F3
      RETURN
      END
C
C
C
      FUNCTION THREDL (X,Y,Z,AX,AY,AZ,AXYZ,NX,NY,NZ)
C
C        THIS FUNCTION INTERPOLATES LINEARLY IN THREE DIMENSIONS.  IT
C     SELECTS THE TWO PLANES OF COSTANT Z AND USES STDLI FOR TWO DIMEN-
C     SIONAL INTERPOLATION.  A LINEAR INTERPOLATION IS USED IN Z.
C        PROGRAMMER  H.SOBEL              DATE  22 AUGUST 1968
      DIMENSION AX(NX), AY(NY), AZ(NZ), AXYZ(NX,NY,NZ)
      DATA M/1/
      CALL SERCH (Z,AZ,K,NZ,M)
      AL=STDLIA(AX,AY,AXYZ(1,1,K),X,Y,NX,NY)
      AU=STDLIA(AX,AY,AXYZ(1,1,K+1),X,Y,NX,NY)
      THREDL=AL+(AU-AL)*(Z-AZ(K))/(AZ(K+1)-AZ(K))
      RETURN
      END
```

LIST OF REFERENCES

1. Honeywell Defense System Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 9-17, 24 March 1978.

2. Honeywell Defense System Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 11, 24 March 1978.

3. Honeywell Defense System Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 21, 24 March 1978.

4. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 1-1.

5. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume II, by John W. Diesel and others, p. C-1-C-37.

6. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume II, by John W. Diesel and others, p. C-38-C-51.

7. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-18.

8. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-16.

9. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-37.

10. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 3-10.

11. Naval Postgraduate School Technical Note 0141-25, User's Guide to Programming Aids Library, by Lloyd G. Nolan, p. 27-29, October 1978.

12. Honeywell Defense Systems Division, System Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, 24 March 1978.

13. Honeywell Defense Systems Division, Systems Design Tradeoff Study For Supersonic Tactical Missile (STM), Volume I, by Robert H. Josselson, p. 77-82, 24 March 1978.

14. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume I, by John W. Diesel and others, p. 4-41-4-83.

15. Litton Systems, Inc., Guidance and Control Systems Division, Modularized Six-Degree-Of-Freedom (MOD6DF) Computer Program, Volume II, by John W. Diesel and Others, p. B-1 - B-49.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
2.	Department Chairman, Code 67 Department of Aeronautical Engineering Naval Postgraduate School Monterey, California 93940	1
3.	Professor D. J. Collins, Code 67CO (thesis advisor) Department of Aeronautical Engineering Naval Postgraduate School Monterey, California 93940	2
4.	LT Thomas Alan Grote, USN (student) 1016 Orchid Avenue Modesto, California 95355	1
5.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2